NAVAL POSTGRADUATE SCHOOL

MONTEREY, CALIFORNIA

THESIS

A VALIDATION OF A MOLECULAR DYNAMICS SIMULATION IN DETERMINING THE THERMAL CONDUCTIVITY OF A LA-ZR PYROCHLORE

by

Jeremiah J. Cheatum

December 2008

Thesis Advisor:	John R. Lloyd
Second Reader:	Young W. Kwon

Approved for public release; distribution is unlimited

THIS PAGE INTENTIONALLY LEFT BLANK

REPORT DOCUMENTATION PAGE			Form Approved OMB No. 0704-0188
Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instruction, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302, and to the Office of Management and Budget, Paperwork Reduction Project (0704-0188) Washington DC 20503.			
1. AGENCY USE ONLY *(Leave blank)*	**2. REPORT DATE** December 2008	**3. REPORT TYPE AND DATES COVERED** Master's Thesis	
4. TITLE AND SUBTITLE A Validation of a Molecular Dynamics Simulation in Determining the Thermal Conductivity of a La-Zr Pyrochlore		**5. FUNDING NUMBERS** N/A	
6. AUTHOR(S) Jeremiah J. Cheatum			
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Naval Postgraduate School Monterey, CA 93943-5000		**8. PERFORMING ORGANIZATION REPORT NUMBER**	
9. SPONSORING /MONITORING AGENCY NAME(S) AND ADDRESS(ES) N/A		**10. SPONSORING/MONITORING AGENCY REPORT NUMBER**	
11. SUPPLEMENTARY NOTES The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government.			
12a. DISTRIBUTION / AVAILABILITY STATEMENT Approved for public release; distribution is unlimited		**12b. DISTRIBUTION CODE A**	

13. ABSTRACT (maximum 200 words)

Semiconductors continue to shrink in size and are now nearing the performance limits of some traditional materials. Silicon Dioxide, which has been used extensively as a gate insulator in MOSFETs, is one such material and so research is focusing on finding a suitable replacement with a high dielectric constant. Oxides of Lanthanum and Zirconium have been identified as possible successors, but these compounds have not been well studied. This thesis is the first step in an attempt to learn more about the thermo-physical and electronic properties of a Lanthanum Zirconium Pyrochlore. A classical molecular dynamics simulation is performed which utilizes a semi-empirical Buckingham interatomic potential to model the van der Waals forces between the atoms in the system. These forces are combined with the electrostatic potential, and the motions of the particles are determined over a corresponding time history. The movement of the energy contained within the atoms is then analyzed using statistical methods to determine the thermal conductivity of the pyrochlore. This conductivity will then be compared with experimental data to determine the validity of the simulation and potential function.

14. SUBJECT TERMS Lanthanum Zirconium Pyrochlore, Classical Molecular Dynamics Simulation, Thermal Conductivity, Gate Insulator, Semiconductors			**15. NUMBER OF PAGES** 97
			16. PRICE CODE
17. SECURITY CLASSIFICATION OF REPORT Unclassified	**18. SECURITY CLASSIFICATION OF THIS PAGE** Unclassified	**19. SECURITY CLASSIFICATION OF ABSTRACT** Unclassified	**20. LIMITATION OF ABSTRACT** UU

NSN 7540-01-280-5500 Standard Form 298 (Rev. 2-89)
Prescribed by ANSI Std. 239-18

THIS PAGE INTENTIONALLY LEFT BLANK

Approved for public release; distribution is unlimited

A VALIDATION OF A MOLECULAR DYNAMICS SIMULATION IN DETERMINING THE THERMAL CONDUCTIVITY OF A LA-ZR PYROCHLORE

Jeremiah J. Cheatum
Lieutenant, United States Navy
B.S., the University of Southern California, 2001

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN MECHANICAL ENGINEERING

from the

NAVAL POSTGRADUATE SCHOOL
December 2008

Author: Jeremiah J. Cheatum

Approved by: John R. Lloyd
Thesis Advisor

Young W. Kwon
Second Reader

Knox T. Millsaps
Chairman, Department of Mechanical and Astronautical Engineering

THIS PAGE INTENTIONALLY LEFT BLANK

ABSTRACT

Semiconductors continue to shrink in size and are now nearing the performance limits of some traditional materials. Silicon Dioxide, which has been used extensively as a gate insulator in MOSFETs, is one such material and so research is focusing on finding a suitable replacement with a high dielectric constant. Oxides of Lanthanum and Zirconium have been identified as possible successors, but these compounds have not been well studied. This thesis is the first step in an attempt to learn more about the thermo-physical and electronic properties of a Lanthanum Zirconium Pyrochlore. A classical molecular dynamics simulation is performed which utilizes a semi-empirical Buckingham interatomic potential to model the van der Waals forces between the atoms in the system. These forces are combined with the electrostatic potential, and the motions of the particles are determined over a corresponding time history. The movement of the energy contained within the atoms is then analyzed using statistical methods to determine the thermal conductivity of the pyrochlore. This conductivity will then be compared with experimental data to determine the validity of the simulation and potential function.

THIS PAGE INTENTIONALLY LEFT BLANK

TABLE OF CONTENTS

THIS PAGE INTENTIONALLY LEFT BLANK

LIST OF FIGURES

THIS PAGE INTENTIONALLY LEFT BLANK

LIST OF TABLES

THIS PAGE INTENTIONALLY LEFT BLANK

ACKNOWLEDGMENTS

I would like to thank my thesis advisor, Dr. John Lloyd for his wisdom and guidance through this project. Even across thousands of miles, he remained a constant source of new ideas.

I would also like to thank Dr. James Luscombe for explaining the finer points of nanoscale and statistical physics in a way that even an engineering student can grasp.

Tengfei Luo has provided many technical insights that have allowed me to solve many of the problems encountered while writing my simulation program and in analyzing the results, and is deserving of much gratitude.

Finally, I need to thank God, who gave me the strength to persevere and to continue to hope. Without these, nothing is possible.

THIS PAGE INTENTIONALLY LEFT BLANK

I. INTRODUCTION

A. SHRINKING SEMICONDUCTORS AND ELECTRON TUNNELLING

In 1975, Gordon Moore stated that the number of semiconductor components able to fit onto an integrated circuit chip could be expected to double every two years [1]. While larger chip areas and reduced defect density certainly contributed to this trend, the major driver has been the increase in density allowed by smaller semiconductors [1]. This process has been continuing steadily since 1959, and begs the question, "Just how long can it last?" As transistor sizes approach the nanoscale level, quantum effects are becoming more significant. Since these effects are not widely understood, especially in how they affect many of the exotic materials that transistors are made out of today, there are abundant research challenges that must be met if the validity of Moore's Law is to continue for the next forty years.

An area in which quantum effects are already causing concern is in relation to gate insulators. In a Metal Oxide Semiconductor Field Effect Transistor (MOSFET), the flow of electrons between the source and the drain is inhibited until a charge is applied to the gate. Once a charge in applied, a channel forms between the source and the drain, and electrons will flow [2]. However, the gate must be separated from the channel by an insulator, as shown in Figure 1. MOSFETs have traditionally used a metal oxide for this insulator and for years, Silicon Dioxide has been the insulator of choice [3].

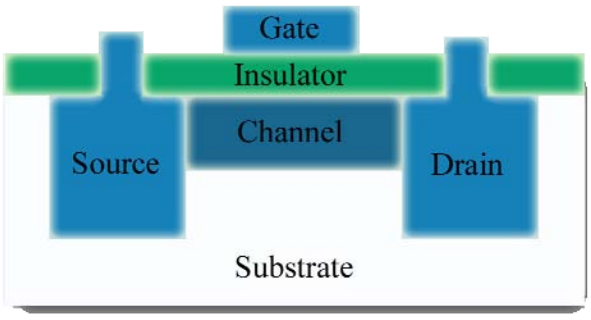

Figure 1. MOSFET Diagram.

MOSFETs are able to be constructed very small, which is also a benefit to manufacturers striving to miniaturize their products. However, the "scaling of dimensions of complimentary metal-oxide-silicon (CMOS) transistors has led to the thickness of the silicon dioxide used as the gate insulator decreasing below 1.6 nm. Below this thickness, the leakage current due to direct tunneling increases above the allowed value of about 1 A/cm^2 [4]." An important aspect of the channel between source and drain is the series capacitance of this channel, which, in a CMOS transistor is controlled by the oxide used as an insulator. "Because of its small dielectric constant, SiO_2 as a gate oxide has emerged as one of the key bottlenecks in device downscaling [5]." Due to this, materials with higher dielectric constants are being sought.

B. SUITABLE REPLACEMENTS

As researchers begin their search for a suitable replacement for silicon dioxide, they are faced with several challenges. The replacement must form a decent bond with silicon, must be thermodynamically stable, must have a high dielectric constant, and must have sufficient band offsets. A category of materials that has caught the interest of researchers are oxides of lanthanum (La), zirconium (Zr), hafnium (Hf), aluminum (Al), and Yttrium (Y) [4]. However, there has not been a lot of research in how to best use these sorts of oxides in semiconductors. In particular, there is a lack of understanding about pyrochlores at microscopic levels, including the material of concern in this study, a lanthanum zirconium pyrochlore [6]. A complication in any research of this kind is that manufacturing MOSFETs which incorporate the oxides in question is a time consuming and expensive process. When the number of requirements that must be met is factored into the search, finding a suitable match becomes exponentially harder. What is needed is a method to cheaply and quickly eliminate possibilities from the list of potential candidates, so that the expensive processes only need to be undertaken for the most promising compounds.

C. THE BENEFIT OF NANOSCALE COMPUTATIONS

This is where Molecular Dynamics Simulations (MDS) offer the most reward. Needing only an interatomic potential and a molecular structure for the material in

question, many of the thermo-physical and electrical properties can be determined. Since many of these materials are not common, and much research remains to be done, MDS studies offer a way for component manufacturers and United States security interests to get ahead of their competition in the world marketplace.

Experiments to determine the molecular structure of many of the oxides mentioned above have already been carried out using transmission electron microscopes. Finding an appropriate interatomic potential is more challenging, as this forms the foundation of any simulation. It is imperative that the potential realistically models the system. Most interatomic potentials are empirical, with parameters matched to the physical properties of the material, such as lattice constant, melting point and elastic constants [7]. These properties can be determined without the need for expensive fabrication techniques and the experiments required are less demanding as well.

For all these reasons, molecular dynamics simulations can fill a vital role in the continuing quest to make smaller, cheaper, and more resilient semiconductors that can be fielded in an ever growing number of areas.

D. A LANTHANUM-ZIRCONIUM PYROCHLORE MD SIMULATION

This paper will investigate the thermal properties of a Lanthanum-Zirconium pyrochlore. The combination of two of the elements that could form a potential gate insulator makes this a likely candidate. This is also one of the few oxides where previous research has identified a suitable interatomic potential, which means that this compound is prime for additional experiments.

The intent of this project is to write an elementary program in the Fortran90 computing language that will utilize the structure and interatomic potential of the La-Zr pyrochlore using a classical Molecular Dynamics Simulation, and that will be capable of determining various thermo-physical and electronic properties using statistical methods. Specifically, the thermal conductivity will be computed and compared with experimental data to determine the validity of the simulation. This program is being written with the expectation that future researchers will expand upon it and incorporate calculations for

other properties and even other materials, and thus continue refining this tool into one that is reliable, robust and applicable to a wide range of nanoscale research.

The La-Zr pyrochlore offers great potential for advancing the science of electronic miniaturization far into the 21st century. This research will further our understanding of the properties of the La-Zr pyrochlore, which will translate directly into application in a wide variety of electronic devices.

II. COMPUTATIONAL MODEL

A. PYROCHLORE STRUCTURE

The La-Zr Pyrochlore consists of ions of lanthanum, zirconium and oxygen. The lanthanum and zirconium cations form a sublattice with a face-centered cubic (FCC) structure. Each atom type is located along the diagonal of three of the sides of the cube, with the other type lining the opposite faces, as shown in Figure 2 [8].

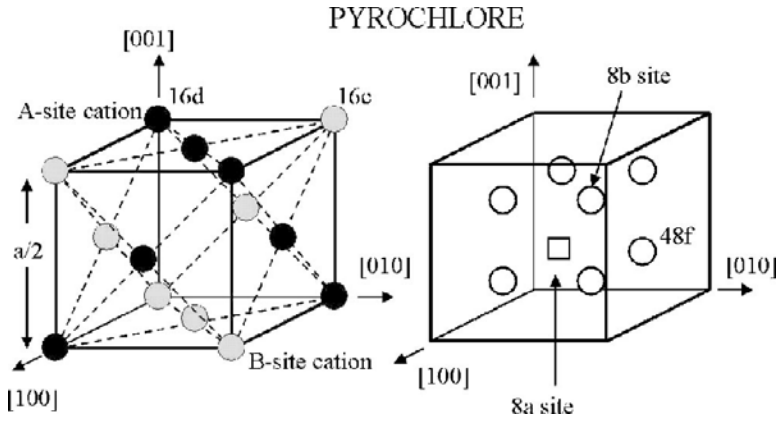

Figure 2. Structure of the La-Zr Pyrochlore. [From [8]]

In this figure, the Zr^{4+} cations are located at the 16c lattice positions (gray), while the La^{3+} cations are located at the 16d positions (black). The O^{2-} anions fill the tetragonal interstitials, with the exception of the 8a site, which is left vacant. It should be noted that the oxygen at the 8b site fills a tetragonal interstitial fully surrounded by lanthanum atoms, while the other interstitials at sites 48f are surrounded by two Lanthanum and two Zirconium atoms. The variation in the charge field caused by the La^{3+} and Zr^{4+} cations, along with the O^{2-} vacancy shifts the locations of these interstitials slightly. This variation is described by a lattice parameter, x = 0.4200 [8].

The pyrochlore unit cell is constructed with the sublattice described above along with a mirror image of it. The positions of the lanthanum and zirconium ions are reversed, and the positions of the oxygen interstitials appropriately adjusted. A group of

eight such sublattices – four of each – arranged in a three dimensional checkerboard pattern makes up one unit cell. The lattice parameter is a = 10.805 Å for the entire unit cell [9].

The main program reads the pyrochlore structure from an input file. The structure was created by combining eight unit cells, with two along each crystallographic direction so that a cubic volume is formed. This was accomplished with a MATLAB code, which is listed in the Appendix, section 1. The code creates an output file with the positions of each atom in the simulation space, its mass and its charge. The output file is then modified so it lists the type of each atom and is formatted properly for a Fortran90 data file. The MATLAB code was written so that the size of the simulation space could be easily changed, simply by specifying the number of unit cells to be included along each crystallographic direction.

B. THE BUCKINGHAM POTENTIAL

An interatomic potential forms the core of any molecular dynamics simulation. It is the governing equation for the interaction of the particles in the simulation. When combined with the electrostatic interactions, it describes the potential energy contained in the system as well as the forces, such as Van der Walls forces and electron cloud interactions that act upon each atom, which determines their motion, and hence their kinetic energy. The interatomic potential used in this study is a type called a Buckingham potential function. A Buckingham potential is a two-body, semi-empirical potential that consists of an exponential term to describe the repulsive potential between particles and an r^{-6} attractive term. The generic Buckingham potential function is shown in Equation (1).

$$U_r = Ae^{\left(\frac{-r}{\rho}\right)} - \frac{C}{r^6} \tag{1}$$

Three fitting constants – A, ρ, and C – are used to match the potential function to the appropriate material by ensuring that the crystallographic data and elastic properties of the material are accurately reproduced by the potential function [8]. The appropriate parameters are listed in Table 1.

Table 1. Interatomic Potential Parameters.

	A (eV)	ρ (Å)	C (eV Å^6)	Mass (u) [10]	Charge (e)
La-O	1367.41	0.35910	0.00	La: 138.90547	La: +3
Zr-O	1478.69	0.35542	0.00	Zr: 91.224	Zr: +4
O-O	22764.30	0.14900	27.89	O: 15.9994	O: -2

In the La-Zr Pyrochlore, the primary interactions between particles are the ionic interactions. The Buckingham potential is used to calculate the short range interactions. As such, parameters are only necessary for the La-O, Zr-O, and O-O interactions. The repulsive coulombic forces between La-La, Zr-Zr, and La-Zr ions are strong enough that the short range forces are negligible [8] and as such, parameters for these interactions were not determined.

The potential function is the source of the force calculation, and they are related as shown in Equation (2).

$$\mathbf{F} = -\frac{\partial U}{\partial r}\hat{\mathbf{r}} \tag{2}$$

C. ELECTROSTATIC FORCES

As an ionic compound, the motion of the La-Zr pyrochlore is largely determined by the coulomb forces. The forces between two ions are described by Equation (3),

$$\mathbf{F}_{ij} = \frac{1}{4\pi\varepsilon_0}\frac{q_i q_j}{r_{ij}^2}\hat{\mathbf{r}}_{ij} \tag{3}$$

where q_i and q_j are the charges of atoms i and j respectively. The permittivity of a vacuum is given by ε_o. The distance between the two particles is given by r_{ij}.

The simulation uses a parameter called the cutoff radius, r_c to determine if atoms are close enough for their interactions to be significant. As long as r_{ij} is within this distance, the force will be calculated. If the two atoms are further apart than this, they are

ignored in order to minimize the processing load. This applies to the Buckingham potential as well. Once all forces between particles close enough to interact is calculated, the net force on each atom is used to calculate the motion of the atom.

1. Charge Neutralization

One of the problems with this approach is that in any spherically truncated volume such as that described above, the net charge within the sphere will usually not be zero. By adjusting the cutoff radius, the charge can be affected, without changing the nature of the physical system. It is obvious that without charge neutrality, a summation of the energy in the system will not converge [11]. Wolf et al. determined that the net charge was localized near the surface of the truncation volume, and proposed that it be assumed to be located exactly at the surface [11]. This allows a charge-neutralizing potential to be introduced at the cutoff radius, which causes the system energy to converge. The corrected force interaction is given in Equation (4).

$$\mathbf{F}_{ij} = \frac{q_i q_j}{4\pi\varepsilon_0}\left(\frac{1}{r_{ij}^2} - \frac{1}{r_c^2}\right)\hat{\mathbf{r}}_{ij} \tag{4}$$

This is identical in execution to the practice of subtracting the force at the cutoff radius to eliminate the discontinuity at this location [12]. Similarly, the same principle is used in the potential energy calculation.

2. System Damping

Wolf also demonstrated that the calculated energy of the system oscillates, depending on the value of the cutoff radius chosen. As the radius was increased, the magnitude of these oscillations decreased, and the system energy stabilized near its experimental value. However, this generally requires a larger cutoff radius than is computationally reasonable [11]. He solved this problem by damping the coulomb pair potential, such that the system energy would be sufficiently precise with a shorter cutoff radius, without changing the fully converged, undamped system energy. This damping is accomplished by distributing the potential energy formula into an error function term,

and a complementary error function term. The derivative of the energy is taken to develop the force equation, which is simplified to Equation (5).

$$\mathbf{F}_{ij} = \frac{q_i q_j}{4\pi\varepsilon_0}\left\{\left(\frac{erfc(\alpha r_{ij})}{r_{ij}^2} + \frac{2\alpha}{\sqrt{\pi}}\frac{e^{(-\alpha^2 r_{ij}^2)}}{r_{ij}}\right) - \left(\frac{erfc(\alpha r_c)}{r_c} + \frac{2\alpha}{\sqrt{\pi}}\frac{e^{(-\alpha^2 r_c^2)}}{r_c}\right)\right\}\hat{\mathbf{r}}_{ij} \qquad (5)$$

The damping parameter α determines how fast the complementary error function decays to zero [11], which sets the cutoff radius necessary for the system to converge. In this study the cutoff radius was set to 10 Å as this was the maximum size that the simulation space allowed. The damping parameter was thus adjusted to give the best results according to the following criteria. The simulation was run with the temperature set to 700K, with values of α adjusted in steps of 0.1 $Å^{-1}$ between 0.2 $Å^{-1}$ and 0.5 $Å^{-1}$. Three data runs were performed at each setting and five points from each run, at increments of 120000, 140000, 160000, 180000 and 198000 time steps, were used to find an average for each run. The average thermal conductivity in each of the three runs was then used to find an overall average. It was determined that setting α = 0.4 $Å^{-1}$ was found to be the best choice for a number of reasons.

When α = 0.4 $Å^{-1}$ the statistical error was minimized, with a standard deviation of 1.19 W/m-K, compared to 1.76 W/m-K for α = 0.3 $Å^{-1}$, and when α = 0.5 $Å^{-1}$, the standard deviation was 4.01 W/m-K. Choosing a damping parameter of 0.4 $Å^{-1}$ balances the effect of the long range atoms on the system energy and conductivity. The coulomb potential is long range, and "in many instances, Coulombic effects seem to cancel almost completely at long range [12]." In a small atomic simulation such as that conducted in this study, however, this long range charge cancelling is not adequately described by the local environment, and can lead to conditionally stable solutions [12]. By damping the coulomb potential, the effect of the longer range atoms is reduced, as would occur in the physical system, and the solution becomes stable. If the damping is too strong, however, the coulombic forces of long range atoms are entirely neglected, and the thermal conductivity determined by the simulation depends only on the local environment, which leads to inconsistent results. This inconsistency between α = 0.4 $Å^{-1}$ and α = 0.5 $Å^{-1}$ is shown in Figures 3 and 4.

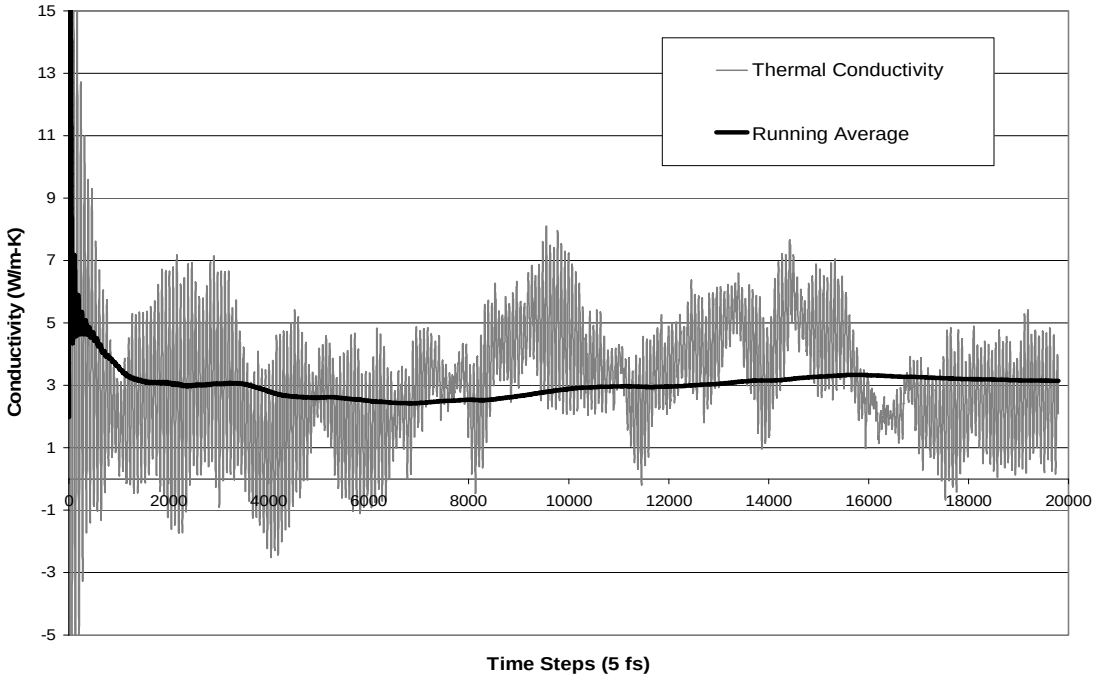

Figure 3. Thermal Conductivity Graph with $\alpha = 0.4$ Å^{-1}.

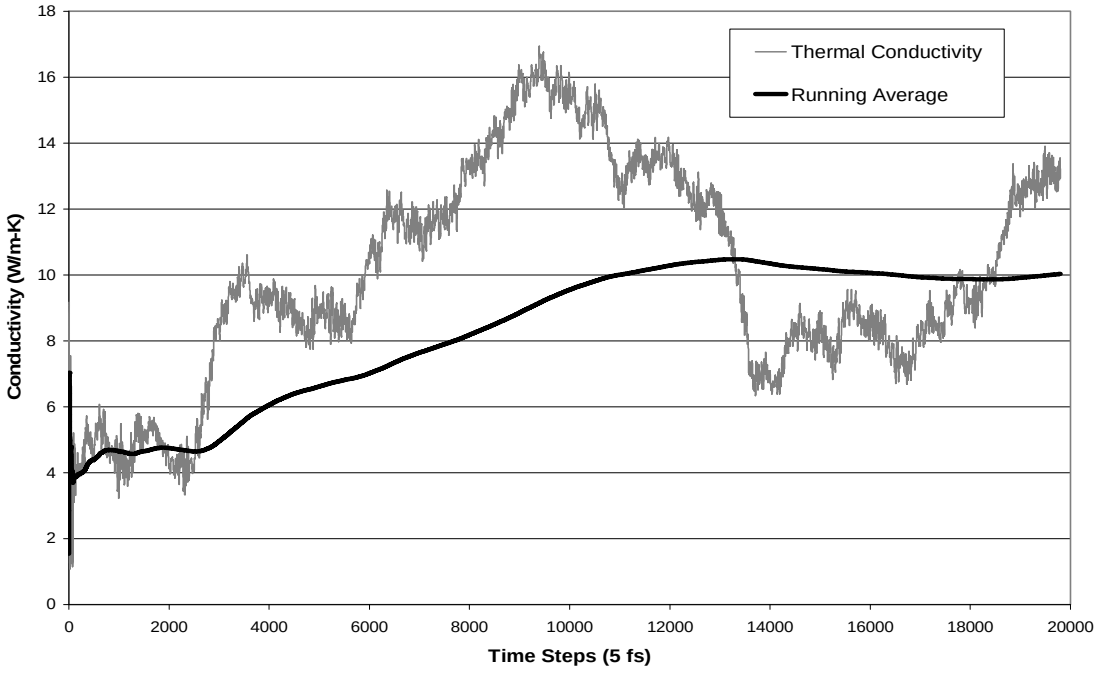

Figure 4. Thermal Conductivity Graph with $\alpha = 0.5$ Å^{-1}.

In these figures, the thermal conductivity is seen to fluctuate over time. This is caused by the presense of optical phonons, and is discussed in more detail in Chapter IV. A running average of the conductivity is calculated to determine a converged thermal conductivity.

Finally, a damping parameter equal to 0.4 $Å^{-1}$ gives a value of the thermal conductivity that matches the experimental bulk value of the conductivity more closely than the other values of alpha. This is shown below in Figure 5, where the thermal conductivity, k = 1.58 W/m-K. This compares well to experimental values of the bulk conductivity, which have determined values of k = 1.52 W/m-K at 700K [13], and k = 2.32 W/m-K at 675K [14]. The value of the Thermal Conductivity at different values of α is shown in Figure 5.

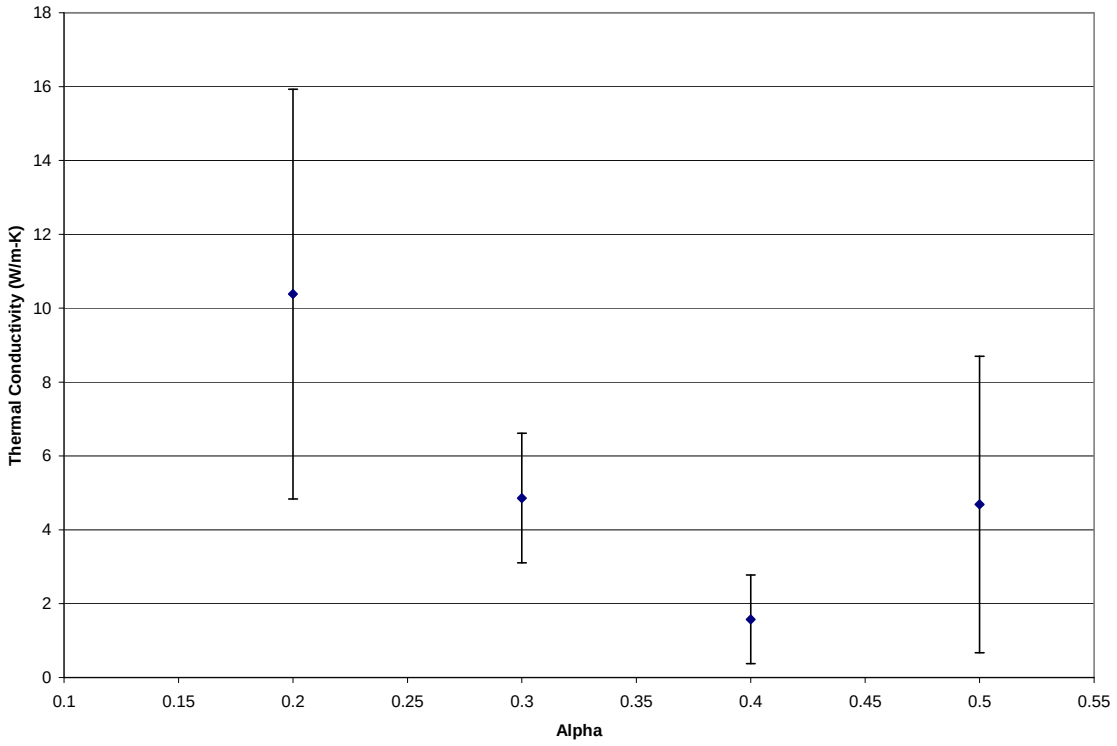

Figure 5. The Affect of α on Thermal Conductivity.

D. STATISTICAL MEASUREMENT OF THE THERMAL CONDUCTIVITY

In order to determine the thermal conductivity, equilibrium statistical methods were utilized. In particular, the thermal conductivity is a transport coefficient, and as such, can be determined through the use of the microscopic autocorrelation function for the heat flux [15].

The Heat Current subroutine of the main program calculates heat current in each crystallographic direction. This data is determined at each time step, and stored until the full length of the simulation has been run. The instantaneous heat fluxes are then used to calculate the Heat Current Autocorrelation Function. Equation (6) gives the thermal conductivity according to the Green-Kubo formula [7].

$$k(T) = \frac{V}{3\kappa_B T^2} \int_0^\infty \overline{\mathbf{J}_Q(t) \bullet \mathbf{J}_Q(0)} dt \tag{6}$$

The integral in Equation (6) samples all possible time origins, and the time average of the system is used in place of the ensemble average since the system is assumed to be ergodic [7]. The time average is computed per Equation (7).

$$\overline{\mathbf{J}_Q(t) \bullet \mathbf{J}_Q(0)} = \frac{1}{M} \sum_{k=1}^{M} \mathbf{J}_Q(t_k) \bullet \mathbf{J}_Q(t_k + t) \tag{7}$$

Equation (7) shows that the time average of the autocorrelation function at each time origin is found taking the dot product of the heat current at the time origin with the heat current of the next M time steps, and finding the average. This integral of these values then gives the conductivity as in Equation (6).

III. COMPUTATIONAL METHOD

A. SIMULATION PROGRAM

The program used for the molecular dynamics simulation, listed in the Appendix, section 2, `DampedPyrochloreIIB.f90`, is based on a code developed by Kim Seng Cheong and Xuan Zheng of the University of Illinois at Urbana-Champaign [16]. They were calculating the thermal conductivity of solid Argon using a Lennard-Jones potential. Their program was modified so that a structure file for any compound could be read, with the only necessary change to the main program being the number of atoms present in the simulation space. The ability to read a parameter input file was also included so that system variables such as temperature, number of time steps, size of each time step, cutoff radius, and quenching intervals could be easily modified. The electrostatic potential, which incorporates charge neutralization and damping, and the Buckingham potential were also implemented in the force calculation subroutine. The program was further modified so that all variables would have proper dimensions, vice using dimensionless quantities. This was done so that the formulas could be checked for dimensional consistency and be easier for the operator to track the series of calculations that were performed. By using the appropriate scales – angstroms, electron volts, femtoseconds, and atomic mass units – the values of most calculations remained close to unity, which is important for high resolution in computer simulations.

The creation of additional output files were also incorporated into the program, so that the motion of the particles, the kinetic, potential and total energy, and their positions could be evaluated. Finally, the ability to store the state of the system at the end of the run was created, with the ability to use this data to initialize a subsequent data run. This proved highly beneficial, in that once the system had reached equilibrium, each subsequent data run started from a state of equilibrium. Since the random motions of the particles quickly erase any memory of the initial state [15], this allows useful measurements to be taken during the majority of each run. For this study, 1000 time steps were allowed at the beginning and end of each simulation run of 300,000 time steps, so that each run would be completely independent.

The program uses periodic boundary conditions. If the motion of a particle takes it outside one of the boundaries, then it is repositioned as if it just entered from the opposite side. The program has been written so that an atom near one (or more) of the boundaries will check the atoms near the opposite boundaries, and include them in its force and energy calculations if they are within the cutoff distance. The program correctly calculates the shortest distance between the atoms by creating an image of the far atom just across the near boundary and measures forces and energies from that image.

B. ATOMIC MOTION

To initialize the simulation, the type, position, mass and charge of each atom are read from the structure file (`pyrostructure.f90` in the Appendix, section 3). The structure file also contains the dimensions of the simulation space. For this study, the x, y, and z dimensions were set to 21.61 Å, which is twice the lattice constant, for a total of eight unit cells. The components of each atom's velocity are then generated from a random number generator that yields a Gaussian distribution.

The components of the momentum of each atom are summed to find the total system momentum, as in Equation (8).

$$p_\alpha = \sum_{i=1}^{N} m_i v_{i\alpha} \tag{8}$$

Each component of momentum is then divided by the total mass, to give the mass normalized velocity of the system. This is subtracted from each atom's individual velocity, which sets the total momentum of the system to zero.

1. Velocities in Subsequent Runs

The option to use data from a previous run has been built into the code.. By setting `runid=0`, the simulation will perform an initial run and assign random velocities to each atom. If the setting is changed to `runid=1`, however, then the atomic positions, mass and charge are read from the `reset` file, which also contains the velocities of each atom.

The momentum of the system is still set to zero, as above, since a slight system drift can develop over time if it is not periodically corrected. This is only done once, at the beginning of each simulation, so that none of the measurements will be affected by a sudden change in atomic motion.

C. TEMPERATURE SCALING

The total kinetic energy of the system is measured throughout the program, as determined in Equation (9).

$$E_{K,Act} = \frac{1}{2}\sum_{i=1}^{N} m_i \left(v_{xi}^2 + v_{yi}^2 + v_{zi}^2 \right) \tag{9}$$

The kinetic energy of the simulation, however, is based off the temperature [7], and is given by

$$E_{K,Spec} = \frac{3}{2} N \kappa_B T \tag{10}$$

where N is the number of atoms in the system, κ_B is the Boltzman constant, and T is the temperature of the system. The kinetic energy is derived from only the three translational degrees of freedom of the atoms in this system. The rotational and vibrational modes are not included since the atoms are modeled as soft spheres. The temperature is specified in the parameter file, `input2.f90`, which is listed in the Appendix, section 4.

For the initial simulation run, the randomly determined velocities give a kinetic energy according to Equation (9), which will not match that obtained via Equation (10). In order to match these energies, a scale factor

$$Scale = \sqrt{\frac{E_{K,Spec}}{E_{K,Act}}} \tag{11}$$

is created, and the velocities are multiplied by it so that the actual kinetic energy of the system will match the kinetic energy required for the simulation to start.

D. EQUILIBRATION

As the first simulation is run, the atoms begin to move into an equilibrium state. As this occurs, their momentum, and thus kinetic energy, will increase. This increased energy is analogous to a higher local temperature, which is the primary driver of mass diffusion in solids [17]. If the difference between the initial positions and the equilibrium positions are extreme, it is possible for the atoms to escape their lattice positions. This is undesirable when the intent for the system is to stabilize in an equilibrium condition, as it takes longer for this to happen. Hence, while the system is equilibrating, the temperature is periodically quenched by applying the scale factor in Equation (11) to the velocities of the atoms in the system. In this study, the temperature was quenched every 100 time steps, for the first 3000 steps of each run. As the system approaches equilibrium, the effect of the quenching is minimized, and finally it is discontinued, as the system must be allowed to come to its final equilibrium position without interference.

E. DETERMINING THE NEIGHBORLIST

A neighborlist is a series of lists, each of which contains the IDs all the atoms within the cutoff distance of any particular atom. They are arranged as a single, long array with a separate pointer array that lists the beginning of the neighborlist for each atom [15]. The subroutine `neighborlist` steps through each atom, calculating the distance between it and all the atoms after it in the atom list. It accounts for the possibility that an atom's image may be closer than the atom, and includes such atoms if they fall within the cutoff radius. It creates the neighborlist array as well as the pointer array. Since each atom is only compared to the atoms that lie after it, there is not any overlap in the neighborlists, i.e., if Atom B is in Atom A's neighborlist, then Atom A will not be in Atom B's list. This minimizes redundant calculations, and speeds up the overall processing of the forces.

F. MOTION CALCULATIONS

The motion of the particles is calculated using the Verlet method. The general flow of this method is that the net force acting on each particle is calculated, and then used to determine a "new" position for the particle at time (t +1). This new position is

compared to an "old" position at time (t – 1) to determine the velocity at time (t). This velocity is used for the energy and heat current calculations. Then the system is stepped forward one time step, so that the current position becomes "old" position, and the "new" position becomes the current one. At this point, if the motion of the particles has taken any of them outside the box, both the current and old positions are moved. By moving the old position as well, the velocity calculation will not be affected.

The first step in this procedure is to use the initial positions and velocities to determine the old position according to Equation (12).

$$\hat{\mathbf{r}}_{i,o} = \hat{\mathbf{r}}_i - tstep \bullet \mathbf{v}_i \tag{12}$$

Next, the program enters into the molecular dynamic loop. At each time step, the program steps through each atom, calculating the forces and potential energy between it (atom *i*) and each atom (atom *j*) in its neighborlist. The distance between atoms *i* and *j* is determined first. Again, the shortest possible distance, based on images across the simulation boundaries is calculated. First, the potential energy is computed.

$$U_{ij} = 14.400 q_i q_j \left(\frac{erfc(\alpha d)}{d} - \frac{erfc(\alpha r_c)}{r_c} \right) + A^* e^{\left(\frac{-d}{\rho^*}\right)} - \frac{C^*}{d^6} \tag{13}$$

The conversion factor *14.400* accounts for the *1/4π* term, the permittivity of a vacuum, and converts the charge to electron volts. The distance between atoms *i* and *j* is labeled by *d*. The cutoff radius is r_c. The parameters of the Buckingham potential depend on what type of atoms *i* and *j* are, and are distinguished by the starred variables above.

For each atom, a self term, shown in Equation (14), must also be included which accounts for the charge of the atom itself.

$$U_{ii} = -14.400 q_i^2 \left(\frac{erfc(\alpha r_c)}{2r_c} + \frac{\alpha}{\sqrt{\pi}} \right) \tag{14}$$

At this point, the potential energy of both atoms *i* and *j*, and the total potential energy is adjusted by U_{ij}. The potential energy at the cutoff radius is subtracted so that continuity is maintained.

The force calculation is performed as per Equation (15). The force between atoms *i* and *j*, is divided by an additional distance term. This is done for computational efficiency.

$$f_r = 14.400 q_i q_j \left[\left(\frac{erfc(\alpha d)}{d^3} + \frac{2\alpha}{\sqrt{\pi}} \frac{e^{\left(-\alpha^2 d^2\right)}}{r_c^2} \right) - \left(\frac{erfc(\alpha r_c)}{r_c^3} + \frac{2\alpha}{\sqrt{\pi}} \frac{e^{\left(-\alpha^2 r_c^2\right)}}{r_c^2} \right) \right] + \frac{A^*}{\rho^* d} e^{\left(\frac{-d}{\rho^*}\right)} - \frac{6C^*}{d^8} \tag{15}$$

With f_r calculated, the components of the force are easily determined from Equation (16),

$$\mathbf{F}_\alpha = f_r \square d\alpha_{ij} \tag{16}$$

where α represents the index **x**, **y**, or **z**, of the direction. This force interaction is then added to the total force acting upon atom *i* and subtracted from the total force on atom *j*.

Once all the forces have been calculated, they are then used to update the coordinates of each atom according to the Verlet algorithm. The formula for the **x** coordinate is given by Equation (17).

$$\mathbf{x}_n = 2\mathbf{x} - \mathbf{x}_o + \frac{\mathbf{F}_x}{m} \Delta t^2 \tag{17}$$

The old position, x_o, the mass of the particle, *m*, and the time step, Δt, are required to compute this new position. It is for this reason that an "old" position for each atom was computed at the beginning of the simulation run, and also why the old position must be moved if the new position is found to be outside the simulation space.

With the new position of the atom determined, the updated velocity can be determined quite simply, and is shown in Equation (18).

$$\mathbf{v}_x = \frac{\mathbf{x}_n - \mathbf{x}_o}{2\Delta t} \tag{18}$$

Once the updated velocities of the atoms have been calculated, the kinetic energy of the system can be recalculated. When combined with the potential energy, the total energy can be determined, and an evaluation of the conservation of energy can be made.

G. THE HEAT CURRENT

The heat current is calculated by computing the motion of the kinetic and potential energy that each particle carries with it. This current is then used to calculate the thermal conductivity by the statistical methods outlined in Section 2.D. In Equation (19), the dot product of the force between atoms *i* and *j* and the velocity of atom *i* is multiplied by the distance between them.

$$\mathbf{J}_Q(t) = \left[\sum_{i=1}^{N} \left(\mathbf{v}_i h_i + \frac{1}{2} \sum_{j=1, j\neq i}^{N} \mathbf{r}_{ij} \left(\mathbf{F}_{ij} \bullet \mathbf{v}_i \right) \right) \right] \tag{19}$$

This expression is decomposed into the components of each coordinate axis, and performed on both atoms simultaneously so that a double summation over all atoms can be reduced to a double summation over just the neighborlist of each atom. The following equations are obtained.

$$fv_\alpha = d\alpha \bullet f_r \left(v_{\alpha,i} + v_{\alpha,j} \right) \tag{20}$$

This computation is performed for each coordinate, α, summed over each coordinate, and then multiplied by the projection of the distance onto the coordinate axis of the x-, y-, and z-directions. This is performed for every interaction, with the results summed as shown in Equation (21).

$$J2_\alpha = \sum_{i=1}^{N} \sum_{j=jnab}^{jend} d\alpha \sum_{\alpha=1}^{3} fv_\alpha \tag{21}$$

The second summation is over the values of *jnab* to *jend*, which are the beginning and end of the neighborlist for each atom. This value of the bond energy, carried by the motion of the particles, is divided by two to give the second term in Equation (19) above.

The first term in Equation (19) is found by multiplying the velocity by the total energy of each particle, which is given by Equation (22).

$$h_i = \frac{p_i^2}{2m} + \sum_{j=1}^{n} u_{ij}\left(r_{ij}\right) \tag{22}$$

The x, y, and z components of the heat flux are calculated at each time step in the simulation run, and stored until the end of the program, for use in calculating the thermal conductivity.

1. Calculating the Thermal Conductivity

With a full set of heat current data, the thermal conductivity can be calculated by applying the Green-Kubo relations discussed above. The first step is to compute the Correlation function. This is done by multiplying the heat flux at each time step by the heat flux at each step after it. For this simulation, 100,000 time steps were deemed sufficient to reach a completely uncorrelated condition. These correlations are then averaged, as shown in Equation (23) to get the heat correlation at time *i*.

$$heatcorr_i = \frac{1}{100,000} \sum_{j=1}^{100,000} \sum_{\alpha=1}^{3} j_\alpha\left(i\right) j_\alpha\left(i+j\right) \tag{23}$$

This was done for 198,000 time steps to find the correlation function for each time step in the program. Equation (23) is essentially a discretised version of Equation (7), which is the computation to find the autocorrelation function. The total autocorrelation function consists of the summation over all *mavg* = 198,000 time steps.

$$J_{ijj} = \sum_{i=1}^{mavg} heatcorr_i \tag{24}$$

To find the thermal conductivity, the value of the Heat Current Auto-Correlation Function found in Equation (24) is entered into Equation (25).

$$k = 1.6022e^6 \frac{J_{ijj}\Delta t}{3\kappa_B V T^2} \tag{25}$$

The leading constant converts the conductivity into W/m-K. It should be noted that this equation differs in form from Equations (6) in that the volume of the simulation space appears in the denominator vice the numerator. This is due to the fact that the heat flux determined in Equation (19) was never divided by the volume in order to reduce the number of arithmetic operations required. Thus, it is actually a heat current, vice heat flux [18]. Due to the heat flux being multiplied by a "future" heat flux in Equation (23), the factor was squared. This is finally corrected in Equation (25) by dividing by the volume instead of multiplying.

THIS PAGE INTENTIONALLY LEFT BLANK

IV. RESULTS AND DISCUSSION

A. DATA PRESENTATION

The simulations were run at temperatures of 200K through 800K in increments of 50K. Each simulation was run for 300,000 time steps, with measurement of the heat current beginning 1000 steps into the run, and ending 1000 steps before the end of the run. It was found that the value of the thermal conductivity varied, depending on the number of steps that it was measured over. This was most likely caused by the presence of optical phonons which introduced high frequency oscillations, so that the heat current auto-correlation function never vanished [18]. This was overcome by computing a running average of the thermal conductivity, where the number of time steps integrated over varied from 1 – 198,000. This calculation was performed by the `HCACAnalyzer.f90` program, which is listed in the Appendix, section 5.

The data from each run was condensed by only using every tenth data point. In order for a run to be considered acceptable it must settle at some steady state value. A maximum variation of ±2 W/m-K was allowed between the 12000^{th} and 19800^{th} point. Most runs were within this limit, but occasionally a run did not stabilize. The results from several of these runs are shown in Figures 6 – 8 below. These graphs show the randomness in the fluctuations in the thermal conductivity over a run. It was more common that a run at a higher temperature would not settle around a steady value.

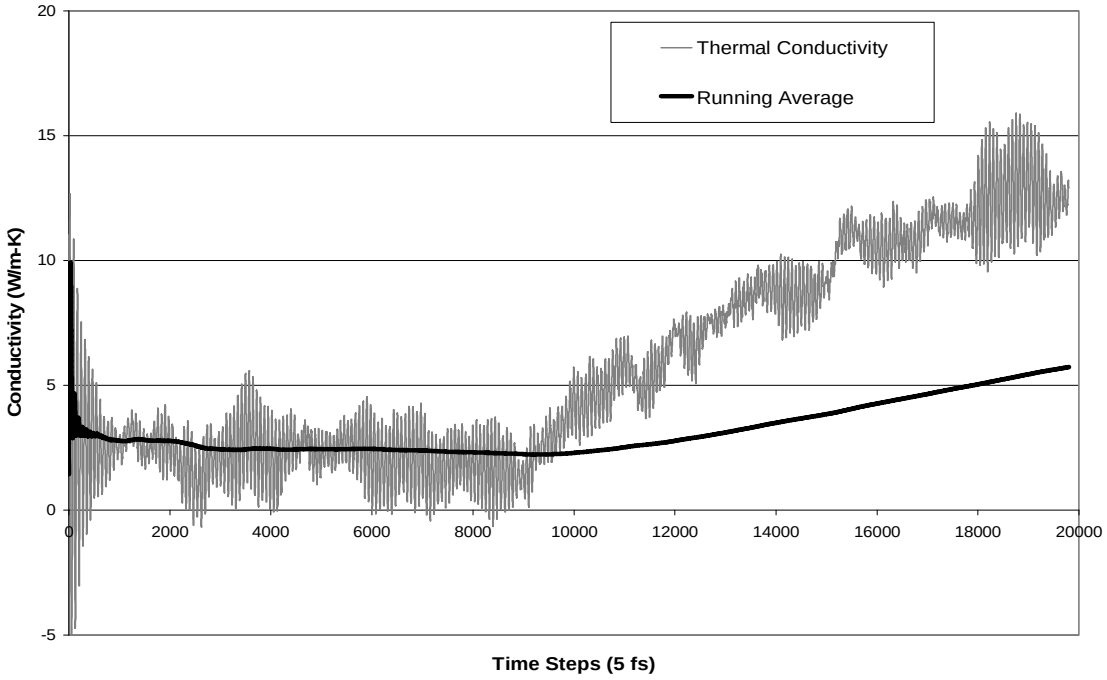

Figure 6. Graph of Unsteady Thermal Conductivity at 550K.

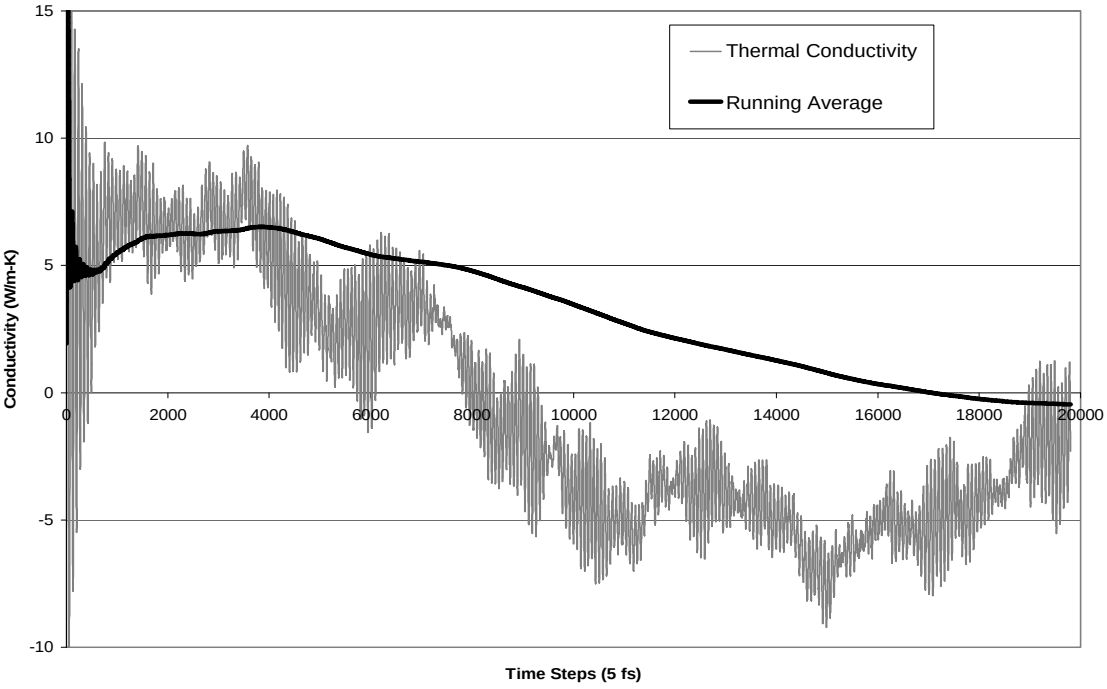

Figure 7. Graph of Unsteady Thermal Conductivity at 700K.

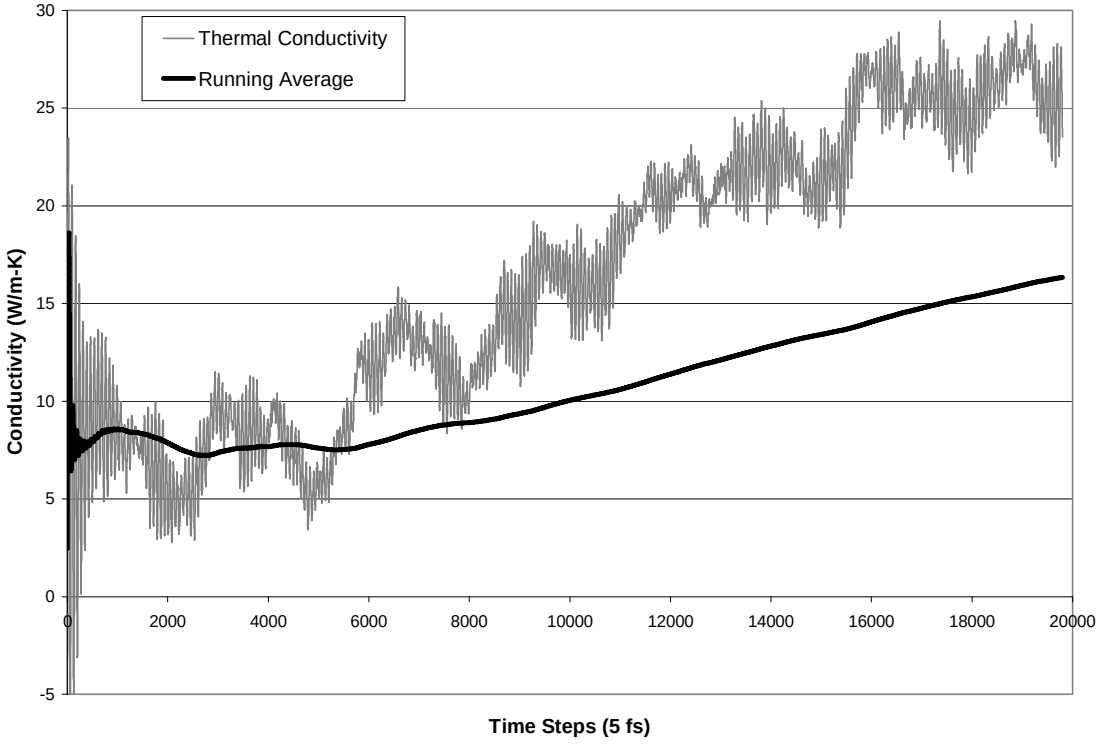

Figure 8. Graph of Unsteady Thermal Conductivity at 800K.

Six data runs were performed at each temperature, so that an average of the thermal conductivity at each temperature could be determined. Five data points from each run were used, at data steps of 12000, 14000, 16000, 18000 and 19800. This was done so that the overall average would more accurately reflect the value each run was steadying at, even if the endpoint had diverged slightly. Below, Tables 2 - 4 show the values of the thermal conductivity at the data steps mentioned above, in each run, for every point in the temperature range.

Table 2. Thermal Conductivity Measurements

200K	**Run 1**	**Run 2**	**Run 3**	**Run 4**	**Run 5**	**Run 6**
12000	0.1818	0.2490	0.3795	0.6421	0.3606	0.1103
14000	0.1753	0.2680	0.3734	0.6383	0.3267	0.0756
16000	0.1600	0.3205	0.3879	0.6242	0.3092	0.0915
18000	0.1268	0.3883	0.3759	0.5998	0.2967	0.0626
19800	0.1050	0.4396	0.3671	0.5568	0.2706	0.0072
250K	**Run 1**	**Run 2**	**Run 3**	**Run 4**	**Run 5**	**Run 6**
12000	0.2597	0.5543	0.1903	0.0726	0.5953	1.0986
14000	0.2179	0.5661	0.1614	0.0900	0.6751	1.0578
16000	0.1875	0.5961	0.1551	0.1210	0.7361	1.0097
18000	0.0953	0.6206	0.1506	0.1763	0.7859	0.9565
19800	0.0210	0.6498	0.1774	0.2205	0.8144	0.9183
300K	**Run 1**	**Run 2**	**Run 3**	**Run 4**	**Run 5**	**Run 6**
12000	4.1192	1.3762	0.7356	1.0664	0.6123	1.1527
14000	4.4290	1.2144	0.7909	1.0263	0.6450	1.0078
16000	4.7192	1.0763	0.8975	0.9140	0.6306	0.9389
18000	5.0631	1.0256	0.9814	0.8071	0.6054	0.9245
19800	5.3512	1.0453	1.0190	0.7980	0.6531	0.9520
350K	**Run 1**	**Run 2**	**Run 3**	**Run 4**	**Run 5**	**Run 6**
12000	1.5058	2.2734	0.3899	1.7303	0.1985	1.1958
14000	1.6858	2.5315	0.3050	1.6465	0.1855	1.3031
16000	1.8662	2.7441	0.3074	1.4476	0.2593	1.3180
18000	2.0821	2.8719	0.3377	1.3271	0.3010	1.2689
19800	2.2485	3.0013	0.3001	1.2679	0.2410	1.2019
400K	**Run 1**	**Run 2**	**Run 3**	**Run 4**	**Run 5**	**Run 6**
12000	4.4389	1.2292	2.2232	1.9144	3.1037	0.6309
14000	4.8194	1.4574	2.4054	1.9229	3.4319	0.6027
16000	5.0633	1.6477	2.5868	1.8390	3.6561	0.5130
18000	5.2962	1.7725	2.9069	1.8447	3.6948	0.4411
19800	5.5387	1.8395	3.2065	1.8478	3.6659	0.3806

Table 3. Thermal Conductivity Measurements (Cont.)

450K	**Run 1**	**Run 2**	**Run 3**	**Run 4**	**Run 5**	**Run 6**
12000	0.7898	0.0076	3.0707	2.5421	2.8805	1.6251
14000	0.5915	-0.3916	3.3881	3.0103	3.1719	2.2114
16000	0.7424	-0.6336	3.6037	3.3778	3.1597	2.7587
18000	0.8594	-0.4577	3.6752	3.7887	3.0464	3.1253
19800	0.9124	-0.1032	3.6836	4.2171	3.0702	3.3326
500K	**Run 1**	**Run 2**	**Run 3**	**Run 4**	**Run 5**	**Run 6**
12000	1.7737	1.1730	1.5786	0.9131	1.0805	0.3879
14000	1.5959	1.1141	1.3652	0.7075	1.2687	0.4438
16000	1.3826	0.8863	1.2545	0.6297	1.2409	0.5626
18000	1.2596	0.7723	1.3599	0.4643	1.1239	0.6579
19800	1.4129	0.6053	1.3281	0.2795	1.0251	0.6046
550K	**Run 1**	**Run 2**	**Run 3**	**Run 4**	**Run 5**	**Run 6**
12000	5.7773	1.9897	2.8458	7.8102	6.3748	4.5598
14000	5.6183	1.7358	2.7261	7.8651	6.5030	4.6654
16000	5.1794	1.5419	2.5083	7.7860	6.9760	4.7568
18000	4.5669	1.3791	2.3489	7.5132	7.5390	4.7543
19800	3.8322	1.2951	2.2016	7.2676	8.1959	4.8919
600K	**Run 1**	**Run 2**	**Run 3**	**Run 4**	**Run 5**	**Run 6**
12000	1.5957	2.1253	1.2775	3.2869	5.1305	6.9593
14000	1.8011	2.0086	0.9470	3.2880	5.3301	7.4700
16000	1.9914	2.0351	0.5731	3.2838	5.2863	8.1114
18000	2.0130	2.0692	0.4694	3.4579	5.1044	8.5503
19800	2.0331	2.0808	0.5362	3.6064	5.0145	8.9178
650K	**Run 1**	**Run 2**	**Run 3**	**Run 4**	**Run 5**	**Run 6**
12000	2.2522	4.9710	3.1970	4.2932	2.4903	3.7477
14000	2.2181	4.8491	3.4495	4.6213	2.2031	3.7133
16000	1.9214	4.7472	3.6358	4.9523	2.0099	4.0600
18000	1.4558	4.7762	3.6402	5.2319	1.8546	4.5362
19800	0.8368	5.1080	3.8659	5.4184	1.4385	4.9023

Table 4. Thermal Conductivity Measurements (Cont.)

700K	**Run 1**	**Run 2**	**Run 3**	**Run 4**	**Run 5**	**Run 6**
12000	2.9601	1.3967	0.8477	5.8774	5.7482	4.5300
14000	3.1541	1.1052	0.7222	5.8874	5.9536	4.4203
16000	3.3243	1.0746	0.4719	5.9234	5.7192	4.2118
18000	3.1976	1.0801	0.2906	6.0507	5.6470	3.7990
19800	3.1415	0.7489	0.1545	5.9663	5.9026	3.4598
750K	**Run 1**	**Run 2**	**Run 3**	**Run 4**	**Run 5**	**Run 6**
12000	4.3849	4.8455	1.8783	0.7700	8.5422	5.9768
14000	4.5009	3.8653	2.0610	0.7436	7.8916	5.3981
16000	4.3677	2.9367	2.4258	0.6090	7.4510	5.1500
18000	4.2688	2.3637	2.5056	0.3897	7.1510	5.2932
19800	4.2138	2.0517	2.5853	0.1716	7.3257	5.4712
800K	**Run 1**	**Run 2**	**Run 3**	**Run 4**	**Run 5**	**Run 6**
12000	7.6631	9.8353	4.7907	7.5283	4.0849	3.0910
14000	7.1933	9.9190	4.3641	7.9128	3.9543	3.1495
16000	6.6686	9.7656	4.4394	7.8435	3.3886	3.2757
18000	6.2594	10.3347	4.6990	7.6980	2.3653	3.4818
19800	5.9639	10.6751	4.9781	7.5908	1.6985	3.7959

Collating this data yields the average thermal conductivity at each temperature, along with its standard deviation and relative error. Table 5 contains these results, while Figure 9 shows them graphically.

Table 5. Statistical Data on the Thermal Conductivity

Temperature	Thermal Conductivity	Standard Deviation	% Error
200K	0.3090	0.1744	56%
250K	0.4644	0.2055	44%
300K	1.5526	1.4708	95%
350K	1.3114	0.8842	67%
400K	2.5307	1.4969	59%
450K	2.1685	1.5075	70%
500K	1.0084	0.4029	40%
550K	4.7668	2.2804	48%
600K	3.5451	2.4877	70%
650K	3.5466	1.3510	38%
700K	3.4256	2.1502	63%
750K	3.9196	2.3758	61%
800K	5.9469	2.5946	44%

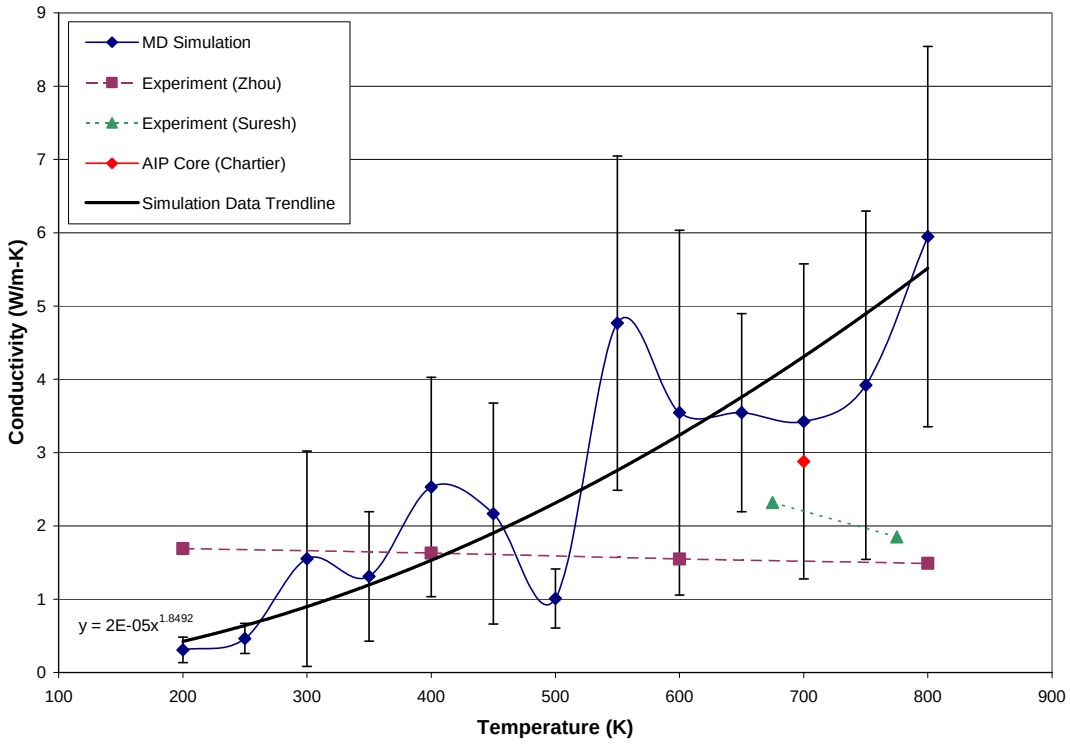

Figure 9. Variation of Thermal Conductivity with Temperature

B. DISCUSSION OF DATA

The first result that becomes apparent is that the simulation is able to model the thermal conductivity to the appropriate order of magnitude. In fact, within the range of 300K to 500K, the conductivity is within ±1 W/m-K of the experimentally determined, bulk value. A typical computer chip operates within a range of 80 - 120°C (353 – 393K) [19], which lies within this range. Thus, this model has shown that the thermal conductivity of a lanthanum zirconium pyrochlore agrees with the experimentally determined bulk value of the thermal conductivity in the range of interest.

A second result that is clearly evident is that the simulation results in a thermal conductivity that grows as the temperature is increased. This behavior does not follow the experimental data which actually decrease slightly as the temperature is raised. A dimensional analysis of the equations used in the simulation shows that the velocity of the particles should increase proportional to the square root of the temperature. The

potential energy and force calculations do not depend on the temperature explicitly – only as the increased motion of the particles causes their positions to become more dynamic, thus changing the distances between particles – does the temperature affect the results. Between 200K and 800K, the difference in the potential energy was less than two percent. The kinetic energy at the most energetic temperature was less than 1.25% of the potential energy. Thus the effect of the temperature on the kinetic and potential energy can be neglected for the dimensional analysis. The heat current is found by multiplying the total energy of each particle, as determined in Equation (22), by the velocity and adding half the bond energy, determined in Equations (20) and (21), as shown in Equation (19). Each term is proportional to the velocity of the particles, and so the heat current is also proportional to the square root of the temperature. In determining the autocorrelation function, the heat current at two separate times is multiplied, which means the autocorrelation function is proportional to the temperature. Finally, to find the thermal conductivity, the autocorrelation function is divided by the square of the temperature, giving a final T^{-1} dependence to the thermal conductivity. This result matches the behavior of the experimental data, and was the expected result of the simulation.

At first, it was thought that the different masses of the particles might be affecting the results. As stated above, the force calculations are largely unaffected by the temperature of the simulation. However, since the force on atom *i* equals the force on atom *j*, the interaction of the two ions will cause a greater acceleration on the oxygen ions than on the heavier lanthanum and zirconium ions. Thus, the oxygen atoms should have a higher velocity. If this velocity is great enough, then the effective temperature of the oxygen atoms would be higher than the average system temperature. This effect has been shown for MD simulations with particles modeled as hard spheres [20]. Since there are 448 oxygen atoms and only 256 of the other two types, this would have a direct impact on the thermal conductivity calculation.

To investigate this, averages of the velocities of each type of atom, as well as the overall average velocity of all atoms were compared. Data was taken from the six runs used to calculate the thermal conductivity at T = 300K and T = 600K. By doubling the

temperature, the velocity would be expected to increase by the square root of two. If the greater average velocity of the oxygen atoms had any affect, then their velocity should increase by greater than the square root of two. The data is given in Table 6.

Table 6. Comparison of the Atomic Velocities

(Å/fs)	**330K**	**600K**	$\mathbf{V_{600}/V_{300}}$
$\mathbf{V_{La}}$	2.16E-03	2.97E-03	1.37
$\mathbf{V_{Zr}}$	2.60E-03	3.80E-03	1.46
$\mathbf{V_{O}}$	6.31E-03	8.88E-03	1.41
$\mathbf{V_{avg}}$	4.88E-03	6.88E-03	1.41

It was found that the velocity of each type of atom actually did scale with temperature, thereby ruling out the possibility that this could be the cause of the rising thermal conductivity. Furthermore, it was found that the ratio of the velocities obeyed the relation given in Equation (26).

$$\frac{\mathbf{v}_2}{\mathbf{v}_1} = \sqrt{\frac{m_1}{m_2}} \tag{26}$$

This result shows that the kinetic energy of each particle is the same, regardless of its mass, and proves that the Equipartition Principle holds true in this simulation. It also shows that the effective temperature of each type of atom is the same as the overall temperature of the system, and thus cannot be the cause of the increasing thermal conductivity with temperature.

Further investigations led to a similar result in a molecular dynamics simulation run by Huang et al. [18], where the thermal conductivity of a MOF-5 crystal also increased. High frequency fluctuations in the heat current autocorrelation function imply the presence of optical phonons which do not have sufficient modes available in a small simulation such as this one to properly model the scattering phenomena which leads to the T^{-1} dependence of the thermal conductivity.

During the investigation into the effect of the temperature on the thermal conductivity, it was noted that the initial oscillations in the running-average thermal conductivity would stabilize at about the same value in each run, and that this would occur before the 1000th data step. It was also noted that this value of the thermal conductivity closely matched the averages obtained from the more complicated procedure detailed above. Not only did it match, but the relative error was greatly reduced. These measurements are displayed in Figure 10.

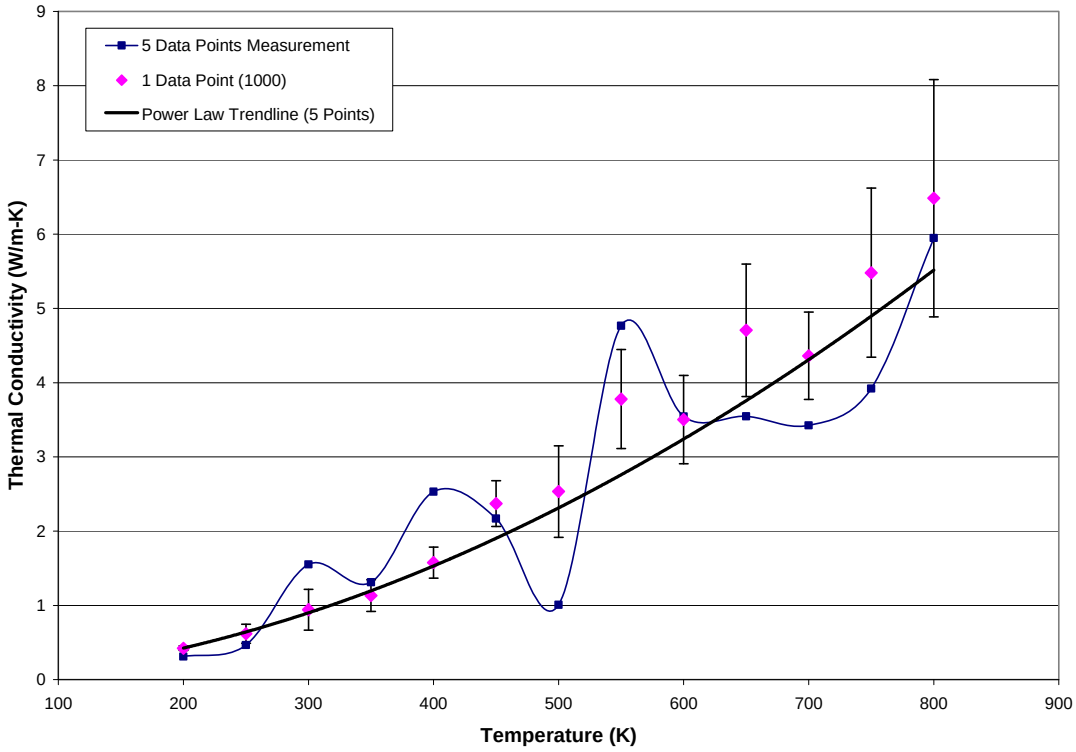

Figure 10. Comparison of Long- and Short-Run Measurement

The explanation for this is that the thermal conductivity actually does converge rather quickly in each simulation. As the simulation is allowed to continue running, however, the random motion of the particles causes the heat current to do one of three things. It can either return to a condition similar to the initial state, in which case, the

calculated thermal conductivity will be higher; move to a state in which the heat flow is generally opposite of that at the beginning of the run, which gives a lower (or even negative) thermal conductivity; or continue toward a completely uncorrelated state, which does not change the thermal conductivity much. Since this is a random process, the averaged value of the thermal conductivity that was determined does not change much, however, the wide range of values that can result, especially at the higher temperatures, causes exceedingly large error margins. This result indicates that a much shorter simulation can be performed, while generating data that is actually more accurate than the longer simulation run provides.

V. CONCLUSION

A. REMARKS ON THIS STUDY

This simulation has demonstrated the ability to determine the thermal conductivity of a lanthanum zirconium pyrochlore in the temperature range that semiconductors normally operate. It was also demonstrated that such underlying concepts as the Equipartition Principle are modeled. This offers great promise that further study into the characteristics of lanthanum zirconium pyrochlores, and other rare earth oxides can be successfully carried out.

The difference in the behavior of the thermal conductivity in this study compared with the experimentally determined values clearly shows the limitations of this study. Without a more thorough understanding of the behavior of phonons, specifically their scattering and absorption, as well as size effects, or perhaps the effects of the boundary conditions, progress in making the simulation more realistic will be limited.

This simulation clearly demonstrates the economy of performing these kinds of simulations. Performing a data run took only six hours on a high performance computing node. With four processors per node, each able to run a simulation, and thirty-two nodes available on the cluster, a series of twelve simulations were able to be run simultaneously, while using only a tenth of the computing power available to the Mechanical Engineering Department. Given the discovery of the fast convergence of the thermal conductivity, this runtime can be reduced by two-thirds. Additional optimization of the code will yield further gains. Clearly, a full diagnostic of a material's properties can be performed in a very short period of time. As more materials are studied, the savings will continue to accumulate as the initial cost of the computer hardware is spread across more simulations.

B. USING LANTHANUM OXIDES IN GATE INSULATORS

There is not enough data to make any conclusive recommendation on the use of a lanthanum zirconium pyrochlore as a gate insulator. The thermal conductivity is rather low – although it is not particularly low when compared to other ceramics – and usually,

materials with low thermal conductivities have low electrical conductivities as well. That is a primary criterion for selection as an advanced gate insulator. Additional study of the characteristics of the lanthanum zirconium pyrochlore, both physical experiments and computer simulations are required, due to the current scarcity of information on this substance.

A lanthanum zirconium pyrochlore is not the only oxide of lanthanum, and there are a variety of dopants that can be added which will change the properties. Modifications to the material being studied can be easily accounted for by this program, thus allowing further study and understanding.

C. DIRECTION OF FUTURE STUDY

This research project was always considered a "proof of concept." It was meant to demonstrate that an interatomic potential could be easily modeled, using a classical molecular dynamics simulation to give meaningful physical properties of a material. By accurately determining the thermal conductivity, it has successfully accomplished this goal. The next step is to expand and build upon what has been learned here.

The first avenue of study is to expand upon the properties that can be calculated by this model. The electrical conductivity, the internal pressure, shear stresses and coefficients of expansion are just a few of the properties that can be determined via computer simulations. A step beyond that would be to incorporate additional potential functions, with the corresponding structure files, in order to investigate other possible gate insulators. As mentioned above, the inclusion of a dopant, even just a few atoms worth, can significantly alter the properties of the material under consideration. The types of materials that can be studied are not just limited to gate insulators, of course.

A more detailed investigation into phonon transport and scattering mechanisms should be undertaken so that these effects can be incorporated into the program's calculations. In particular, it is possible that the arrangement of the lanthanum and zirconium atoms may result in some anisotropic affects that could be exploited by semiconductor manufacturers. The effects of different simulation sizes and geometries on the phonon transport should also be explored. Given the application of interest, a

simulation space that mimicked a thin film, with the appropriate boundary conditions would most likely prove to be highly beneficial.

Improving the processing of the program is an area that also warrants additional study. The number of simulation steps that must be performed, as well as the appropriate time to make a measurement, are both areas that could be optimized. The availability of parallel processing resources almost requires that an effort to optimize and streamline this program be embarked upon.

THIS PAGE INTENTIONALLY LEFT BLANK

APPENDIX. COMPUTER PROGRAMS

1. MATLAB STRUCTURE CREATION FILE

```
% structurecreator will create the locations of the atoms in a
% structure file for my thesis.
% This version creates a La2Zr2O7 pyrochlore structure.  (2/28/07)

clear,clc

% Parameters
a0 = 10.805 % Lattice Parameter (Angstroms)
g = 0  % counter

% Basic Cell - The numbers below are the miller indices in [h k l]
% format.

% Prime Sublattice 1
% Zirconium Locations
Zr1(1,:) = [0 0 0];  % Lattice Point
Zr1(2,:) = [.25 .25 0];
Zr1(3,:) = [.25 0 .25];
Zr1(4,:) = [0 .25 .25];
% Oxygen Locations
% 48f Tetrahedral Sites
Ox1(1,:) = [.42 .125 .125];
Ox1(2,:) = [.125 .42 .125];
Ox1(3,:) = [.375 .375 .17];
Ox1(4,:) = [.125 .125 .42];
Ox1(5,:) = [.375 .17 .375];
Ox1(6,:) = [.17 .375 .375];
% 8b tetrahedral
Ox1(7,:) = [.375 .375 .375];

% Prime Sublattice 2
% Zirconium Locations
La1(1,:) = [0 0 0];  % Lattice Point
La1(2,:) = [.25 .25 0];
La1(3,:) = [.25 0 .25];
La1(4,:) = [0 .25 .25];
% Oxygen Locations
% 48f Tetrahedral Sites
Ox1(8,:) = [.33 .125 .125];
Ox1(9,:) = [.375 .375 .08];
Ox1(10,:) = [.125 .33 .125];
Ox1(11,:) = [.125 .125 .33];
Ox1(12,:) = [.375 .08 .375];
Ox1(13,:) = [.08 .375 .375];
% 8b tetrahedral
Ox1(14,:) = [.125 .125 .125];


% Unit Cell construction - The following section will create the total
```

```
% unit cell based upon the combination of the prime sublattices.
for i = 1:4
    Zr(i,:) = Zr1(i,:);
    Zr(i+4,:) = [.5 .5 0] + Zr1(i,:);
    Zr(i+8,:) = [.5 0 .5] + Zr1(i,:);
    Zr(i+12,:) = [0 .5 .5] + Zr1(i,:);
    La(i,:) = [.5 0 0] + La1(i,:);
    La(i+4,:) = [0 .5 0] + La1(i,:);
    La(i+8,:) = [0 0 .5] + La1(i,:);
    La(i+12,:) = [.5 .5 .5] + La1(i,:);
end
for i = 1:7
    Ox(i,:) = Ox1(i,:);
    Ox(i+7,:) = [.5 0 0] + Ox1(i,:);
    Ox(i+14,:) = [.5 .5 0] + Ox1(i,:);
    Ox(i+21,:) = [0 .5 0] + Ox1(i,:);
    Ox(i+28,:) = [.5 0 .5] + Ox1(i+7,:);
    Ox(i+35,:) = [0 0 .5] + Ox1(i+7,:);
    Ox(i+42,:) = [0 .5 .5] + Ox1(i+7,:);
    Ox(i+49,:) = [.5 .5 .5] + Ox1(i+7,:);
end

% Convert lattice positions to dimensions (in angstroms).
La = La.*a0
Zr = Zr.*a0
Ox = Ox.*a0

% The number of unit cells in each of the [hkl] directions.
a = 2;
b = 2;
c = 2;

% Create vectors for use in the shift matrix.
m = [1 0 0];
n = [0 1 0];
p = [0 0 1];

% Loop for adding additional unit cells
for i = 0:(a-1)
    for j = 0:(b-1)
        for k = 0:(c-1)
            shift = a0*(i*m + j*n + k*p)
            for f = 1:16
                La(16*g+f,:) = shift + La(f,:);
                Zr(16*g+f,:) = shift + Zr(f,:);
            end
            for f = 1:56
                Ox(56*g+f,:) = shift + Ox(f,:);
            end
            g = g + 1
        end
    end
end

% Add Mass and Charge to each Atom
```

```
La(:,4) = 138.90547  % Atomic Mass (g/mol)
La(:,5) = 3  % Charge
Zr(:,4) = 91.224
Zr(:,5) = 4
Ox(:,4) = 15.9994
Ox(:,5) = -2

% Print output for copying into Fortran90 structure file
La
Zr
Ox
```

2. MAIN PROGRAM

```
    PROGRAM DampedPyrochlore

! Purpose:
! Compute the thermal conductivity of a Lanthanum Zirconium Pyrochlore.

! DampedPyrochloreII added the capability to output the final state of
! the system, and begin another run from this state. (10/6/08)

! Programmed J(0)dotJ(0) for the exponential decay of the auto-
! correlation function. (10/17/08)

! Fixed Motion bug - now every run has the motion of the center reset
! to zero. (10/24/08)

    IMPLICIT none

! Declare variables

! ad: alpha*distance between atoms.
! alpha: Damping Parameter
! alpha2: alpha^2
! arcut: alpha*rcut
! boxx, boxy, boxz: length of each side of the box.(angstroms)
! charge(i): the electostatic charge of each atom.
! d: distance between atoms.
! d2: d^2
! di: inverse of d
! d2i: inverse of d^2
! d6i: inverse of d^6
! dlow: minimum distance between atoms
! dhigh: maximum distance between atoms
! dx, dy, dz: distance between the coordinates of atoms.
! e(i): energy of atom i.
! ecut: potential energy at the cutoff radius.
! erfd: the error function of the distance between atoms.
! erfrcut: the error function of the cutoff radius.
! fcur: variable name for the heat current file.
! finitial: variable name for the initial conditions file.
! fkin: variable name for the energy file.
! flambda: variable name for the thermal conductivity file.
```

```
! fmove: variable name for the motion of the center of mass.
! fpar: variable name for the input/parameter file.
! fpos: variable name for the Initial/Final atom positions file.
! freset: variable name for the continuing input structure file.
! freset1: variable name for the output structure file.
! fstruct: variable name of the structure file.
! ftrbl: variable name of the troubleshooting file.
! fvel: variable name of the velocity file.
! fr: force between atoms divided by the distance between them.
! fvx, fvy, fvz: power of the atomic motion. (eV/fs)
! fx(i), fy(i), fz(i): force on each atom along x, y, z directions.
! heatcorr: Heat Current Auto-Correlation Function
! i, j, n: counters
! i2tstep: inverse of 2*tstep.
! iboxx, iboxy, iboxz: inverse of the simulation dimensions.
! ihigh: number of the atom with the highest velocity.
! ilow: number of the atom with the lowest velocity.
! imsvonv: inverse of msconv.
! initx(i), inity(i), initz(i): Initial positions of each atom.
! ipi: inverse of pi.
! j2x, j2y, j2z: the potential energy portion of the heat current
! jbeg: beginning of the neighborlist for each atom.
! jend: end of the neighborlist for each atom.
! jijj: The aggregate heat current autocorrelation function.
! jj: J(0) -dot- J(0)
! jjlambda: The thermal conductivity, based on jjtotal.
! jnab: counter for the atoms within neighborlist of current atom.
! jjtotal: The HCAC function as a sum of jj over all time origins.
! jx, jy, jz: the heat current returned by the subroutine
! jxi, jyi, jzi: heat current at step i
! kb: Boltzmann's Constant -  (eV/K)
! kin(10000): kinetic energy of the system at each write_scalar steps.
! lambda:  Thermal Conductivity
! list: A list of atom IDs which form the neighborlist for each atom.
! mavg: A counter used in the heat current subroutine.
! msconv: Conversion factor - 1.0364D2 (eV-fs^2/u-Ang^2)
! masst: total system mass
! maxnab: an array length for the neighborlist.
! ms(i): mass of atom i (u)
! mvx0, mvy0, mvz0: system momentum in each direction.
! mvx0t, mvy0t, mvz0t: system momentum components, scaled for temp.
! natm: number of atoms in the box.
! natmspec: The number of each type of atom.
! nequ: the time step when the system is in equilibrium.
! nlist: list of atom IDs which form the neighborlist for each atom.
! nquench: The number of times the program has been quenched.
! nrgin: specified initial system energy, based on temperature (eV)
! nscalar: The interval at which output data is stored.
! nsp: identifies the type of atom read from the structure file.
! nspec: The number of different types of atoms.
! nstep: total number of steps.
! qstep: A counter until the next quench.
! quench_interval: The number of steps between quenches.
! quench_times: maximum number of times the program can be quenched.
! parA1,A2,A3,B1,B2,B3,C1,C2,C3: Parameters of the Buckingham potential
```

```
! point: array that points to the neighborlist for the current atom.
! pot(i): potential energy of each atom. (eV)
! potential: potential energy of the system at each write_scaler steps.
! qstep: a step counter within each quench interval.
! quench_interval: Number of steps between quenches
! quench_times: maximum number of quenches.
! rangauss: random number generator - returns a gaussian distribution.
! rcut: Cutoff radius. (Angstroms)
! rcuti: inverse of rcut.
! rcut2: rcut^2.
! rcut2i: inverse of rcut^2.
! rcut6i: inverse of rcut^6.
! runid: differentiates the initial run from subsequent ones.
! scale: The factor that converts system energy to specified energy.
! shiftx, shifty, shiftz: shift amount if an atom leaves the box.
! specname(i): The chemical symbol of each atom.
! spectype(i): An identifier of each atoms type.
! step: the current time step.
! total energy of the system. (eV)
! temp: initial temperature. (K)
! temp2: Temperature squared.
! tenergy: total energy at each time step. (eV)
! tstep: time step size (5.0D-16 sec, 0.5 femto-seconds)
! tstep2: tstep^2
! uij: potential energy between atoms i and j.
! ukin: kinetic energy of the system at each step. (eV)
! upot: potential energy of the system at each step. (eV)
! v2: velocity squared.
! vhigh: velocity of the fastest atom at each time step.
! vlow: velocity of the slowest atom at each time step.
! volume: Volume of the system
! vx(i), vy(i), vz(i): velocity components of each atom. (m/s)
! write_scalar: intervals between energy output data.
! x(i), y(i), z(i): position of each atom. (angstroms)
! xi, yi, zi: position of the current atom. (angstoms)
! xo(i), yo(i), zo(i): old positon of each atom. (angstroms)
! xn(i), yn(i), zn(i): new position of each atom. (angstroms)


    COMMON /block1/ x, y, z
    COMMON /block2/ xo, yo, zo
    COMMON /block3/ vx, vy, vz, ukin, ms
    COMMON /block4/ fx, fy, fz, upot, alpha
    COMMON /block5/ boxx, boxy, boxz, step, tstep
    COMMON /block6/ rcut, point, list, spectype
    COMMON /block7/ pot, j2x, j2y, j2z, charge
    COMMON /block8/ qstep, nquench, quench_times, quench_interval
    COMMON /block9/ temp, runid

    INTEGER :: natm, i, j, n, nsp, nspec, nstep, step, write_scalar,&
     nscalar, quench_interval, quench_times, nquench, &
     qstep, maxnab, runid
    PARAMETER (natm=704, maxnab=140800, nspec=3)
    INTEGER :: point(natm), mavg, nequ, list(maxnab), spectype(natm)
    INTEGER :: natmspec(nspec)
```

```
      PARAMETER (mavg=198000, nequ=1000)
      DOUBLE PRECISION :: boxx, boxy, boxz, ms(natm), charge(natm)
      DOUBLE PRECISION :: tstep, temp, msconv, alpha, &
       initx(natm), inity(natm), initz(natm), &
       x(natm), y(natm), z(natm), &
       vx(natm), vy(natm), vz(natm), &
       fx(natm), fy(natm), fz(natm), &
       xo(natm), yo(natm), zo(natm), &
       mvx0, mvy0, mvz0, mvx0t, mvy0t, mvz0t, &
       v2, vlow, vhigh, rcut, upot, ukin, &
       pot(natm), j2x, j2y, j2z, jx, jy, jz, &
       jxi(5000000), jyi(5000000), jzi(5000000), &
       kb, masst, nrgin, scale, rangauss, &
       jijj, volume, lambda, jj, jjtotal, jjlambda, &
       heatcorr(mavg), kin(100000), potential(100000), tenergy(100000)
      PARAMETER (kb=8.6173D-5) !(eV/K)
      PARAMETER (msconv=1.0364D2) ! (eV-fs^2/u-Ang^2)
      CHARACTER :: fcur*7, finitial*7, fkin*6, flambda*4, fmove*6, &
       fpar*10, fpos*9, freset*5, freset1*6, fstruct*18, ftrbl*15, &
       fvel*8, specname(natm)*2

      runid = 1
      alpha = 0.4D0

  ! Variable names for the output files are assigned.
      fcur='current'                          ! UNIT = 13
      finitial='initial'                      ! UNIT = 202
      fkin='energy'                           ! UNIT = 10
      flambda='HCAC'                          ! UNIT = 11
      fmove='motion'                          ! UNIT = 16
      fpar='input2.f95'                       ! UNIT = 9
      fpos='positions'                        ! UNIT = 12
      freset='reset'                          ! UNIT = 17
      freset1='reset1'                        ! UNIT = 18
      fstruct='pyrostructure.f95'             ! UNIT = 201
      ftrbl='troubleshooting'                 ! UNIT = 14
      fvel='velocity'                         ! UNIT = 15

      OPEN (UNIT=9,FILE=fpar,STATUS='OLD')
      READ (9,*) rcut
      WRITE (*,*) rcut
      READ (9,*) temp
      WRITE (*,*) temp
      READ (9,*) nstep
      WRITE (*,*) nstep
      READ (9,*) tstep
      WRITE (*,*) tstep
      READ (9,*) quench_interval
      WRITE (*,*) quench_interval
      READ (9,*) quench_times
      WRITE(*,*) quench_times
      READ(9,*) write_scalar
      WRITE(*,*) write_scalar

      nquench = 0
```

```
    step = 0
    qstep = 0

! Read size of the system from the input file.
    OPEN (UNIT=201,FILE=fstruct,STATUS='OLD')
    READ (201,*) boxx, boxy, boxz
    WRITE(*,*) boxx, boxy, boxz
    OPEN (UNIT=202,FILE=finitial,STATUS='unknown')
    WRITE (202,*) "*******************************"
    WRITE (202,*) "STRUCTURE FILE: "
    WRITE (202,902) 'BOX DIMENSIONS:',boxx, boxy, boxz

    DO j=1,nspec
      natmspec(j)=0
    ENDDO

    natmspec(1)=1
    nsp=1

! Initial Positions
    IF (runid .EQ. 0) THEN
      DO i=1,natm
        READ (201,*) specname(i), x(i), y(i), z(i), ms(i), charge(i)
        WRITE (202,901) specname(i), x(i), y(i), z(i), ms(i), charge(i)
      ENDDO
    ELSEIF (runid .NE. 0) THEN
      DO i=1,natm
        OPEN (UNIT=17,FILE=freset,STATUS='OLD')
        READ (17,*) specname(i), x(i), y(i), z(i), ms(i), charge(i)
        READ (17,*) vx(i), vy(i), vz(i)
      ENDDO
    ENDIF

    masst = 0.0D0

    DO i=1,natm
      initx(i) = x(i) ! Stores the initial positions of all the atoms
      inity(i) = y(i) ! for comparison to the final positions at the
      initz(i) = z(i) ! end of the run.
      masst = masst + ms(i)

! Specify atom type
      IF (i.EQ.1) THEN
        spectype(nsp)=1
      ELSEIF (specname(i).EQ.specname(i-1)) THEN
        natmspec(nsp)=natmspec(nsp)+1
        spectype(i)=nsp
      ELSEIF (specname(i).NE.specname(i-1)) THEN
        nsp=nsp+1
        natmspec(nsp)=natmspec(nsp)+1
        spectype(i)=nsp
      ENDIF
    ENDDO

    WRITE(*,*) 'Total mass (u):',masst
```

```
    DO j=1,nspec
      WRITE(202,*) 'species',j,'  has',natmspec(j),'  atoms.'
    ENDDO

! Setup initial velocities, & computes system mass & momentum.
    mvx0 = 0.0D0
    mvy0 = 0.0D0
    mvz0 = 0.0D0
    mvx0t = 0.0D0
    mvy0t = 0.0D0
    mvz0t = 0.0D0

    IF(runid .EQ. 0) THEN
! The program assigns random initial velocities.
      DO i=1,natm
! units set to (Ang/fs) by conversion factors below.
        vx(i) = rangauss()
        vy(i) = rangauss()
        vz(i) = rangauss()
      ENDDO
    ENDIF
    DO i=1,natm
      mvx0 = mvx0 + vx(i)*ms(i) ! (u-Ang/fs)
      mvy0 = mvy0 + vy(i)*ms(i)
      mvz0 = mvz0 + vz(i)*ms(i)
    ENDDO

! Reset system momentum to zero.
    mvx0 = mvx0/masst ! Creates mass normalized system velocity.
    mvy0 = mvy0/masst
    mvz0 = mvz0/masst

    ukin = 0.0D0
    vhigh = 0.0D0

    DO i=1,natm
      vx(i) = vx(i) - mvx0 ! Total momentum of the system = 0.
      vy(i) = vy(i) - mvy0
      vz(i) = vz(i) - mvz0
      v2 = vx(i)*vx(i) + vy(i)*vy(i) + vz(i)*vz(i)
      ukin = ukin + ms(i)*v2 ! (u-Ang^2/fs^2)
    ENDDO
! Calculate the kinetic energy of the system with the random velocities
    ukin = 0.5D0*msconv*ukin ! (eV)
    OPEN (UNIT=10,FILE=fkin,STATUS='unknown')
    WRITE(10,*) 'Initial random Kinetic Energy =',ukin,'eV.'
! Calculate the specified system energy, based on the input temperature
    nrgin = 1.5D0*DBLE(natm)*kb*temp
    WRITE(10,*) 'Initial specified Kinetic Energy =',nrgin,'eV.'
    scale = DSQRT(nrgin/ukin) ! Calculates the scale factor.

! Scales the velocities based on temperature.
    ukin = 0.0D0
    DO i=1,natm
```

```
      vx(i)=scale*vx(i)  ! (Ang/fs)
      vy(i)=scale*vy(i)
      vz(i)=scale*vz(i)
      v2 = vx(i)*vx(i) + vy(i)*vy(i) + vz(i)*vz(i)
      ukin = ukin + ms(i)*v2
      IF(i .EQ. 1) THEN
        vlow = v2
        ELSEIF(v2 .LT. vlow) THEN
        vlow = v2
      ELSEIF(v2 .GT. vhigh) THEN
        vhigh = v2
      ENDIF
      mvx0t = mvx0t + vx(i)*ms(i) ! (amu-Ang/fs)
      mvy0t = mvy0t + vy(i)*ms(i)
      mvz0t = mvz0t + vz(i)*ms(i)
    ENDDO

    ukin = 0.5D0*msconv*ukin ! (eV)
    WRITE (10,*) 'Corrected Initial Kinetic Energy =',ukin,'eV.'

    vlow = DSQRT(vlow)
    vhigh = DSQRT(vhigh)

    mvx0t = mvx0t/masst ! (Ang/fs) Average system velocity.
    mvy0t = mvy0t/masst
    mvz0t = mvz0t/masst

    OPEN (UNIT=14,FILE=ftrbl,STATUS='unknown')

    OPEN (UNIT=15,FILE=fvel,STATUS='unknown')
    WRITE (15,*) 'Velocities at ', step
    vlow = 0.0D0
    vhigh = 0.0D0
    DO i=1,natm
      v2 = vx(i)*vx(i) + vy(i)*vy(i) + vz(i)*vz(i)
      v2 = DSQRT(v2)
      IF (i .EQ. 1) THEN
        vlow = v2
      ELSEIF (v2 .LT. vlow) THEN
        vlow = v2
      ELSEIF (v2 .GT. vhigh) THEN
        vhigh = v2
      ENDIF
      WRITE (15,906) i, vx(i), vy(i), vz(i), v2
    ENDDO
    WRITE (15,*) 'Minimum Velocity is ',vlow
    WRITE (15,*) 'Maximum Velocity is ',vhigh

    OPEN (UNIT=16,FILE=fmove,STATUS='unknown')
    WRITE (16,*) 'Initial velocity of the center is', &
     mvx0t, mvy0t, mvz0t
    WRITE (*,*) 'Initial velocity of the center is', &
     mvx0t, mvy0t, mvz0t

! Calcuate the previous position using the generated velocity.
```

```
    DO i=1,natm
      xo(i)=x(i)-tstep*vx(i)
      yo(i)=y(i)-tstep*vy(i)
      zo(i)=z(i)-tstep*vz(i)
    ENDDO
    WRITE (*,*) 'Initialization done.'

!   ************************ MD LOOP ***************************
    WRITE(*,*) 'MD loop.....'

    CALL neighborlist
    WRITE(*,*) 'Neighborlist Generated'

    DO i=1,nstep
      step = step + 1
      qstep = qstep + 1

      CALL force

      CALL heatcurrent(jx, jy, jz)
      jxi(i)=jx ! (eV-Ang/fs)
      jyi(i)=jy
      jzi(i)=jz

      CALL integrate

      nscalar = (step/write_scalar) + 1
      kin(nscalar) = ukin ! (eV)
      potential(nscalar) = upot
      tenergy(nscalar) = ukin + upot

! Checks the movement of the center of mass at each time step,
! after the new velocities are calculated.
      mvx0t = 0.0D0
      mvy0t = 0.0D0
      mvz0t = 0.0D0
      DO n=1,natm
        mvx0t = mvx0t + vx(n)*ms(n) ! (amu-Ang/fs)
        mvy0t = mvy0t + vy(n)*ms(n)
        mvz0t = mvz0t + vz(n)*ms(n)
      ENDDO
      mvx0t = mvx0t/masst ! (Ang/fs) Average system velocity.
      mvy0t = mvy0t/masst
      mvz0t = mvz0t/masst
      WRITE (16,905) step, mvx0t, mvy0t, mvz0t

!  Check the velocities when the system energy spikes.
      IF (ukin .GT. 200.0D0) THEN
        WRITE (15,*) 'Velocities at ', step
        vlow = 0.0D0
        vhigh = 0.0D0

        DO n=1,natm
          v2 = vx(n)*vx(n) + vy(n)*vy(n) + vz(n)*vz(n)
          v2 = DSQRT(v2)
```

```
          IF (n .EQ. 1) THEN
            vlow = v2
          ELSEIF (v2 .LT. vlow) THEN
            vlow = v2
          ELSEIF (v2 .GT. vhigh) THEN
            vhigh = v2
          ENDIF
          WRITE (15,906) n, vx(n), vy(n), vz(n), v2
        ENDDO

        WRITE (15,*) 'Minimum Velocity is ',vlow
        WRITE (15,*) 'Maximum Velocity is ',vhigh
      ENDIF
    ENDDO

!   ************************ MD LOOP DONE ************************
    WRITE(*,*) 'MD loop done.'

! Calculate the thermal conductivity.
    jijj=0.0D0
    jjtotal = 0.0D0

    DO i=1,mavg
      jj = jxi(i)*jxi(i) + jyi(i)*jyi(i) + jzi(i)*jzi(i)
      jjtotal = jjtotal + jj
      heatcorr(i) = 0.0D00
      DO j=nequ,(nequ + 100000)
        heatcorr(i) = heatcorr(i) + jxi(i+j)*jxi(j) &
         + jyi(i+j)*jyi(j) + jzi(i+j)*jzi(j)
      ENDDO
      heatcorr(i) = heatcorr(i)/(100000)
      jijj = jijj + heatcorr(i) ! (eV^2-Ang^2/fs^2)
    ENDDO
    jjtotal = jjtotal/mavg

    WRITE (*,*) 'Heat correlation function generated.'
    WRITE(*,*) jijj

    volume = boxx*boxy*boxz
    lambda = 1.6022D6*jijj*tstep/3.0D0/kb/volume/temp/temp ! W/m-K
    jjlambda = 1.6022D6*jjtotal*tstep/3.0D0/kb/volume/temp/temp

    WRITE(*,*) 'thermal conductivity', lambda

    OPEN (UNIT=11,FILE=flambda,STATUS='unknown')
    WRITE(11,*) 'temperature =', temp, 'K'
    WRITE(11,*) 'thermal conductivity =', lambda, 'W/m-K'
    WRITE(11,*) 'K(w) =', jjlambda, 'W/m-K'
    WRITE(11,*) 'Alpha =', alpha
    DO i=1, mavg
      WRITE (11, *) i, heatcorr(i)
    ENDDO

    DO i=1,nscalar
      WRITE (10,*)  i, kin(i), potential(i), tenergy(i)
```

```
    ENDDO
    OPEN (UNIT=12,FILE=fpos,STATUS='unknown')
    WRITE(12,*) natm
    WRITE(12,*) boxx, boxy, boxz
    DO i=1, natm
      WRITE(12,*) 'Initial Positions - Final Positions'
      WRITE(12,903) i, initx(i), inity(i), initz(i), x(i), y(i), z(i)
    ENDDO

    OPEN (UNIT=13,FILE=fcur,STATUS='unknown')
    DO i=nequ,(nequ+mavg)
      WRITE (13,*) i, jxi(i)
    ENDDO

    OPEN (UNIT=18,FILE=freset1,STATUS='unknown')
    DO i=1,natm
      WRITE(18,901) specname(i), x(i), y(i), z(i), ms(i), charge(i)
      WRITE(18,907) vx(i), vy(i), vz(i)
    ENDDO

901   FORMAT(A4,3X,F10.5,3X,F10.5,3X,F10.5,3X,F10.5,3X,F10.5)
902   FORMAT(A16,3X,F15.10,3X,F15.10,3X,F15.10,3X,F15.10,3X,F15.10)
903   FORMAT(I3,3X,F15.10,3X,F15.10,3X,F15.10,3X,F15.10,3X, &
       F15.10,3X,F15.10)
905   FORMAT(I3,1X,ES15.8,1X,ES15.8,1X,ES15.8)
906   FORMAT(I3,1X,ES14.7,1X,ES14.7,1X,ES14.7,1X,ES14.7)
907   FORMAT(F10.5,3X,F10.5,3X,F10.5)

    END
! End of the main program.



    SUBROUTINE neighborlist
! purpose:
! Calculates the neighborlist at the start of each run. Because it is
! a solid, the neighbor list does not need to be updated.

    IMPLiCIT none

    COMMON /block1/ x, y, z
    COMMON /block5/ boxx, boxy, boxz, step, tstep
    COMMON /block6/ rcut, point, list, spectype

    INTEGER natm, maxnab, step
    PARAMETER (natm=704, maxnab=140800)
    INTEGER point(natm), list(maxnab), spectype(natm)
    DOUBLE PRECISION x(natm), y(natm), z(natm), &
     boxx, boxy, boxz, tstep, rcut
    DOUBLE PRECISION xi, yi, zi, dx, dy, dz, d2, &
     iboxx, iboxy, iboxz, rcut2
    INTEGER i, j, nlist

    iboxx = 1.0D0/boxx
    iboxy = 1.0D0/boxy
```

```
    iboxz = 1.0D0/boxz
    rcut2=(rcut+0.1D0)*(rcut+0.1D0)

    nlist=0

    WRITE(*,*) 'Generating neighbor list.......'

    DO i=1,(natm-1)
      point(i)=nlist+1
      xi=x(i)
      yi=y(i)
      zi=z(i)
! WRITE(*,*) 'neighbor list'
      DO j=i+1,natm
        dx=xi-x(j)
        dy=yi-y(j)
        dz=zi-z(j)
        dx=dx-DNINT(dx*iboxx)*boxx
        dy=dy-DNINT(dy*iboxy)*boxy
        dz=dz-DNINT(dz*iboxz)*boxz
        d2=dx*dx+dy*dy+dz*dz
        IF (d2 .LT. rcut2) THEN
          nlist=nlist+1
          list(nlist)=j
        ENDIF
      ENDDO
    ENDDO

! WRITE (*,*) 'Neighbor list done'
    point(natm)=nlist+1
    END


    SUBROUTINE force

! Purpose:
! To calculate the force, the potential energy, and the heat current.

    IMPLICIT none

    COMMON /block1/ x, y, z
    COMMON /block3/ vx, vy, vz, ukin, ms
    COMMON /block4/ fx, fy, fz, upot, alpha
    COMMON /block5/ boxx, boxy, boxz, step, tstep
    COMMON /block6/ rcut, point, list, spectype
    COMMON /block7/ pot, j2x, j2y, j2z, charge

    INTEGER natm, i, j, jbeg, jend, jnab, maxnab, step
    PARAMETER (natm=704, maxnab=140800)
    INTEGER point(natm), list(maxnab), spectype(natm)
    DOUBLE PRECISION x(natm), y(natm), z(natm), xi, yi, zi
    DOUBLE PRECISION fx(natm), fy(natm), fz(natm), charge(natm)
    DOUBLE PRECISION vx(natm), vy(natm), vz(natm), ms(natm)
    DOUBLE PRECISION upot, ukin, uij, fr, ecut(3), tstep
    DOUBLE PRECISION dx, dy, dz, d, di, d2, d2i, d6i, dlow, dhigh
```

```
    DOUBLE PRECISION boxx, boxy, boxz, iboxx, iboxy, iboxz
    DOUBLE PRECISION pot(natm), j2x, j2y, j2z, fvx, fvy, fvz
    DOUBLE PRECISION parA1, parB1, parC1, parA2, parB2, parC2, &
     parA3, parB3, parC3
    DOUBLE PRECISION rcut, rcuti, rcut2, rcut2i, rcut6i
    DOUBLE PRECISION alpha, alpha2, ipi, arcut, erfrcut, ad, erfd
! La-O Parameters
    PARAMETER (parA1=1.36741D3, parB1=2.7847396D0, parC1=0.0D0)
! Zr-O Parameters
    PARAMETER (parA2=1.47869D3, parB2=2.8135727D0, parC2=0.0D0)
! O-O Paramters
    PARAMETER (parA3=2.27643D4, parB3=6.7114094D0, parC3=27.89D0)
    PARAMETER (ipi=0.564189583548)

    alpha2 = alpha*alpha
    rcuti = 1.0D0/rcut
    rcut2=rcut*rcut
    rcut2i=rcuti*rcuti
    rcut6i=rcut2i*rcut2i*rcut2i
    iboxx=1.0D0/boxx
    iboxy=1.0D0/boxy
    iboxz=1.0D0/boxz

    dlow = 0.0D0
    dhigh = 0.0D0

! Potential energy at cutoff radius due to the Buckingham potential.
    ecut(1) = parA1*exp(-rcut*parB1) - parC1*rcut6i ! (eV)
    ecut(2) = parA2*exp(-rcut*parB2) - parC2*rcut6i ! (eV)
    ecut(3) = parA3*exp(-rcut*parB3) - parC3*rcut6i ! (eV)

! Set forces, potential energy and pressure to zero.
    DO i=1,natm
      fx(i)=0.0D0
      fy(i)=0.0D0
      fz(i)=0.0D0
      pot(i)=0.0D0
    ENDDO

    upot=0.0D0

    j2x=0.0D0
    j2y=0.0D0
    j2z=0.0D0

! Compute the error function for the cutoff radius.
    arcut = rcut*alpha
    CALL error(arcut,erfrcut)

! Compute forces and energies for each atom.
    DO i=1,(natm-1)
      jbeg=point(i)
      jend=point(i+1)-1

      IF (jbeg .LE. jend) THEN
```

```
        xi=x(i)
        yi=y(i)
        zi=z(i)

        DO jnab=jbeg,jend
          j=list(jnab)
!  write(*,*) j

! Potential energy at cutoff radius.
          dx = xi - x(j)
          dy = yi - y(j)
          dz = zi - z(j)

          dx = dx - DNINT(dx*iboxx)*boxx
          dy = dy - DNINT(dy*iboxy)*boxy
          dz = dz - DNINT(dz*iboxz)*boxz

          d2 = dx*dx + dy*dy + dz*dz
          d = DSQRT(d2)

! Compute the error function of the damped distance between atoms.
          ad = alpha*d
          CALL error(ad,erfd)

          IF (i .EQ. 1 .AND. jbeg .EQ. point(1)) THEN
            dlow = d
          ELSEIF (d .LT. dlow) THEN
            dlow = d
          ELSEIF (d .GT. dhigh) THEN
            dhigh = d
          ENDIF

          IF (d2 .LT. rcut2) THEN
            di=1.0D0/d
            d2i=di*di
            d6i=d2i*d2i*d2i

! *****  Potential & Force Calculations  *****
! 1.4399938D1 converts the energy to eV.
! units for fr are (eV/Ang^2)
            IF (spectype(i) .EQ. 1) THEN  ! La
             IF (spectype(j) .EQ. 3) THEN  ! O
              uij = 1.44D1*charge(i)*charge(j)*((1 - erfd)*di &
               - (1 - erfrcut)*rcuti) + parA1*DEXP(-d*parB1) &
               - parC1*d6i
              IF (j .EQ. jbeg) THEN
               uij = uij - 1.44D1*charge(i)*charge(i)*(0.5D0*(1 &
                - erfrcut)*rcuti + alpha*ipi)
              ENDIF
              upot = upot + uij - ecut(1)
              pot(i) = pot(i) + uij - ecut(1)
              pot(j) = pot(j) + uij - ecut(1)
              fr = 1.44D1*charge(i)*charge(j)*(((1 - &
               erfd)*d2i*di + 2*alpha*ipi*DEXP(-alpha2*d2)*d2i) &
               - ((1 - erfrcut)*rcut2i*rcuti + &
```

```
   2*alpha*ipi*DEXP(-alpha2*rcut2)*rcut2i)) + &
   (parA1*parB1*di)*DEXP(-d*parB1) - 6.0D0*parC1*d6i*d2i
  ELSE
   uij = 1.44D1*charge(i)*charge(j)*((1 - erfd)*di &
    - (1 - erfrcut)*rcuti)
   IF (j .EQ. jbeg) THEN
    uij = uij - 1.44D1*charge(i)*charge(i)*(0.5D0*(1 &
    - erfrcut)*rcuti + alpha*ipi)
   ENDIF
  upot = upot + uij
  pot(i) = pot(i) + uij
  pot(j) = pot(j) + uij
  fr = 1.44D1*charge(i)*charge(j)*(((1 - &
   erfd)*d2i*di + 2*alpha*ipi*DEXP(-alpha2*d2)*d2i) &
   - ((1 - erfrcut)*rcut2i*rcuti + &
   2*alpha*ipi*DEXP(-alpha2*rcut2)*rcut2i))
  ENDIF
 ELSEIF (spectype(i) .EQ. 2) THEN  ! Zr
  IF (spectype(j) .EQ. 3) THEN  ! O
   uij = 1.44D1*charge(i)*charge(j)*((1 - erfd)*di &
    - (1 - erfrcut)*rcuti) + parA2*DEXP(-d*parB2) &
    - parC2*d6i
   IF (j .EQ. jbeg) THEN
    uij = uij - 1.44D1*charge(i)*charge(i)*(0.5D0*(1 &
     - erfrcut)*rcuti + alpha*ipi)
   ENDIF
   upot = upot + uij - ecut(2)
   pot(i)=pot(i)+uij-ecut(2)
   pot(j)=pot(j)+uij-ecut(2)
   fr = 1.44D1*charge(i)*charge(j)*(((1 - &
    erfd)*d2i*di + 2*alpha*ipi*DEXP(-alpha2*d2)*d2i) &
    - ((1 - erfrcut)*rcut2i*rcuti + &
    2*alpha*ipi*DEXP(-alpha2*rcut2)*rcut2i)) + &
    (parA2*parB2*di)*DEXP(-d*parB2) - 6.0D0*parC2*d6i*d2i
  ELSE
   uij = 1.44D1*charge(i)*charge(j)*((1 - erfd)*di &
    - (1 - erfrcut)*rcuti)
   IF (j .EQ. jbeg) THEN
    uij = uij - 1.44D1*charge(i)*charge(i)*(0.5D0*(1 &
     - erfrcut)*rcuti + alpha*ipi)
   ENDIF
   upot = upot + uij
   pot(i) = pot(i) + uij
   pot(j) = pot(j) + uij
   fr = 1.44D1*charge(i)*charge(j)*(((1 - &
    erfd)*d2i*di + 2*alpha*ipi*DEXP(-alpha2*d2)*d2i) &
    - ((1 - erfrcut)*rcut2i*rcuti + &
    2*alpha*ipi*DEXP(-alpha2*rcut2)*rcut2i))
  ENDIF
 ELSEIF (spectype(i) .EQ. 3) THEN  ! O
  IF (spectype(j) .EQ. 3) THEN  ! O
   uij = 1.44D1*charge(i)*charge(j)*((1 - erfd)*di &
    - (1 - erfrcut)*rcuti) + parA3*DEXP(-d*parB3) &
    - parC3*d6i
   IF (j .EQ. jbeg) THEN
```

```
                uij = uij - 1.44D1*charge(i)*charge(i)*(0.5D0*(1 &
                 - erfrcut)*rcuti + alpha*ipi)
               ENDIF
               upot = upot + uij - ecut(3)
               pot(i) = pot(i) + uij - ecut(3)
               pot(j) = pot(j) + uij - ecut(3)
               fr = 1.44D1*charge(i)*charge(j)*(((1 - &
                erfd)*d2i*di + 2*alpha*ipi*DEXP(-alpha2*d2)*d2i) &
                - ((1 - erfrcut)*rcut2i*rcuti + &
                2*alpha*ipi*DEXP(-alpha2*rcut2)*rcut2i)) + &
                (parA3*parB3*di)*DEXP(-d*parB3) - 6.0D0*parC3*d6i*d2i
              ELSE
               uij = 1.44D1*charge(i)*charge(j)*((1 - &
                erfd)*di - (1 - erfrcut)*rcuti)
               IF (j .EQ. jbeg) THEN
                uij = uij - 1.44D1*charge(i)*charge(i)*(0.5D0*(1 &
                 - erfrcut)*rcuti + alpha*ipi)
               ENDIF
               upot = upot + uij
               pot(i) = pot(i) + uij
               pot(j) = pot(j) + uij
               fr = 1.44D1*charge(i)*charge(j)*(((1 - &
                erfd)*d2i*di + 2*alpha*ipi*DEXP(-alpha2*d2)*d2i) &
                - ((1 - erfrcut)*rcut2i*rcuti + &
                2*alpha*ipi*DEXP(-alpha2*rcut2)*rcut2i))
              ENDIF
             ENDIF

             fx(i) = fx(i) + fr*dx ! (eV/Ang)
             fy(i) = fy(i) + fr*dy
             fz(i) = fz(i) + fr*dz

             fx(j) = fx(j) - fr*dx
             fy(j) = fy(j) - fr*dy
             fz(j) = fz(j) - fr*dz

! Second term of j(t).
             fvx = dx*fr*(vx(i) + vx(j)) ! (eV/fs)
             fvy = dy*fr*(vy(i) + vy(j))
             fvz = dz*fr*(vz(i) + vz(j))
             j2x = j2x + dx*(fvx + fvy + fvz) ! (eV-Ang/fs)
             j2y = j2y + dy*(fvx + fvy + fvz)
             j2z = j2z + dz*(fvx + fvy + fvz)

           ENDIF
         ENDDO
       ENDIF
     ENDDO


     WRITE (14,*) step, dlow, dhigh
     END
```

```
    SUBROUTINE heatcurrent(jx, jy, jz)

! Purpose:
! to calculate the heat current.

    IMPLICIT none

    COMMON /block3/ vx, vy, vz, ukin, ms
    COMMON /block7/ pot, j2x, j2y, j2z, charge

    INTEGER natm, i, maxnab
    DOUBLE PRECISION msconv, ukin, j2x, j2y, j2z, jx, jy, jz, v2
    PARAMETER (natm=704, maxnab=140800, msconv=1.0364D2)
    DOUBLE PRECISION vx(natm), vy(natm), vz(natm), pot(natm), &
     e(natm), charge(natm), ms(natm)

    jx=0.0D0
    jy=0.0D0
    jz=0.0D0

    DO i=1,natm
      e(i)=0.0D0
    ENDDO

    DO i=1,natm
      v2 = vx(i)*vx(i) + vy(i)*vy(i) + vz(i)*vz(i) ! (Ang/fs)^2
      e(i)=0.5D0*msconv*ms(i)*v2 + 0.5D0*pot(i) ! (eV)

      jx = jx + vx(i)*e(i) ! (eV-Ang/fs)
      jy = jy + vy(i)*e(i)
      jz = jz + vz(i)*e(i)
    ENDDO

    jx = jx + 0.5D0*j2x ! (eV-Ang/fs)
    jy = jy + 0.5D0*j2y
    jz = jz + 0.5D0*j2z

    END


    SUBROUTINE integrate

!Purpose:
!To integrate the equation of motion using Verlet algorithm.

    IMPLICIT none

    COMMON /block1/ x, y, z
    COMMON /block2/ xo, yo, zo
    COMMON /block3/ vx, vy, vz, ukin, ms
    COMMON /block4/ fx, fy, fz, upot, alpha
    COMMON /block5/ boxx, boxy, boxz, step, tstep
    COMMON /block8/ qstep, nquench, quench_times, quench_interval
    COMMON /block9/ temp, runid
```

```
      INTEGER i, natm, nquench, qstep, quench_times, step, runid
      INTEGER quench_interval, maxnab
      DOUBLE PRECISION msconv, imsconv, tstep, tstep2, i2tstep, scale, &
       boxx, boxy, boxz, iboxx, iboxy, iboxz, shiftx, shifty, shiftz
      DOUBLE PRECISION temp, ukin, upot, nrgin, kb, alpha
      PARAMETER (natm=704, maxnab=140800)
      PARAMETER (msconv=1.0364D2, kb=8.6173D-5)
      DOUBLE PRECISION x(natm), y(natm), z(natm), ms(natm)
      DOUBLE PRECISION xo(natm), yo(natm), zo(natm)
      DOUBLE PRECISION fx(natm), fy(natm), fz(natm)
      DOUBLE PRECISION xn(natm), yn(natm), zn(natm)
      DOUBLE PRECISION vx(natm), vy(natm), vz(natm)

      ukin=0.0D0
      tstep2=tstep*tstep
      i2tstep=0.5D0/tstep
      imsconv = 1.0D0/msconv

      iboxx = 1.0D0/boxx
      iboxy = 1.0D0/boxy
      iboxz = 1.0D0/boxz

! Integrate the equation of motion using Verlet algorithm.

      DO i=1, natm
        xn(i) = 2.0D0*x(i) - xo(i) + imsconv*fx(i)*tstep2/ms(i) ! (Ang)
        yn(i) = 2.0D0*y(i) - yo(i) + imsconv*fy(i)*tstep2/ms(i)
        zn(i) = 2.0D0*z(i) - zo(i) + imsconv*fz(i)*tstep2/ms(i)

        vx(i) = (xn(i) - xo(i))*i2tstep ! (Ang/fs)
        vy(i) = (yn(i) - yo(i))*i2tstep
        vz(i) = (zn(i) - zo(i))*i2tstep

        ukin = ukin + ms(i)*(vx(i)*vx(i) + vy(i)*vy(i) + vz(i)*vz(i))
      ENDDO
      ukin = 0.5D0*msconv*ukin

! For warm up steps, use velocity scaling to get the exact temperature
      IF(qstep .EQ. quench_interval .AND. nquench .LE. quench_times) THEN
        nrgin = 1.5D0*kb*DBLE(natm)*temp
        scale=DSQRT(nrgin/ukin)
        qstep=0
        nquench=nquench+1
      ELSE
        scale = 1.0D0
      ENDIF

      ukin = 0.0D0

      DO i=1, natm
! Scale the velocity of the atoms to the initial temperature.
        vx(i)=scale*vx(i)
        vy(i)=scale*vy(i)
        vz(i)=scale*vz(i)
```

```
      xn(i) = x(i) + vx(i)*tstep
      yn(i) = y(i) + vy(i)*tstep
      zn(i) = z(i) + vz(i)*tstep

      ukin=ukin + ms(i)*(vx(i)*vx(i) + vy(i)*vy(i) + vz(i)*vz(i))

      xo(i)=x(i)
      yo(i)=y(i)
      zo(i)=z(i)

      x(i)=xn(i)
      y(i)=yn(i)
      z(i)=zn(i)

! Put the atoms back into the box. If the new positons are moved,
! the old positions must be moved too.

      IF (x(i) .GT. boxx) THEN
        shiftx = DINT(x(i)*iboxx)*boxx
        x(i) = x(i) - shiftx
        xo(i) = xo(i) - shiftx
      ELSEIF (x(i) .LT. 0.0D0) THEN
        shiftx = DINT(x(i)*iboxx - 1.0D0)*boxx
        x(i)=x(i) - shiftx
        xo(i)=xo(i) - shiftx
      ENDIF

      IF (y(i) .GT. boxy) THEN
        shifty = DINT(y(i)*iboxy)*boxy
        y(i)=y(i) - shifty
        yo(i)=yo(i) - shifty
      ELSEIF (y(i) .LT. 0.0D0) THEN
        shifty = DINT(y(i)*iboxy - 1.0D0)*boxy
        y(i)=y(i) - shifty
        yo(i)=yo(i) - shifty
      ENDIF

      IF (z(i) .GT. boxz) THEN
        shiftz = DINT(z(i)*iboxz)*boxz
        z(i)=z(i) - shiftz
        zo(i)=zo(i) - shiftz
      ELSEIF (z(i) .LT. 0.0D0) THEN
        shiftz = DINT(z(i)*iboxz - 1.0D0)*boxz
        z(i)=z(i) - shiftz
        zo(i)=zo(i) - shiftz
      ENDIF
    ENDDO
    ukin = 0.5D0*msconv*ukin
    END


    SUBROUTINE ERROR(X,ERR)
```

```
! Purpose:
! Compute error function erf(x)

! Input:   x   --- Argument of erf(x)
! Output:  ERR --- erf(x)

    IMPLICIT NONE

    INTEGER k
    DOUBLE PRECISION eps, pi, x, x2, er, err, r, c0
    PARAMETER (eps=1.0D-15, pi=3.141592653589793D0)

    x2 = x*x
    IF (DABS(x) .LT. 3.5D0) THEN
      er = 1.0D0
      r = 1.0D0
      DO 10 k=1,50
        r = r*x2/(DBLE(k) + 0.5D0)
        er = er + r
        IF (DABS(r) .LE. DABS(er)*eps) GO TO 15
10    CONTINUE
15    c0 = 2.0D0/DSQRT(pi)*x*DEXP(-x2)
      err = c0*er
    ELSE
      er = 1.0D0
      r = 1.0D0
      DO 20 k=1,12
        r = -r*(DBLE(k) - 0.5D0)/x2
20      er = er + r
        c0 = DEXP(-x2)/(DABS(x)*DSQRT(pi))
        err = 1.0D0 - c0*er
        IF (x .LT. 0.0D0) err = -err
    ENDIF
    RETURN

    END


    FUNCTION ran_uniform()

! Purpose:
! To generate a serial of uniformly distributed random numbers
! with the seed idum.

    IMPLICIT none

    INTEGER idum
    DOUBLE PRECISION ran_uniform,ran2

    SAVE idum
    DATA idum/-10/

    ran_uniform = ran2(idum)

    RETURN
```

```
      END


      FUNCTION ran2(idum)

      IMPLICIT none

      INTEGER idum,im1,im2,imm1,ia1,ia2, &
       iq1,iq2,ir1,ir2,ntab,ndiv

      DOUBLE PRECISION ran2,am,eps,rnmx

      PARAMETER (im1=2147483563,im2=2147483399, &
       am=1.0d0/im1,imm1=im1-1,ia1=40014, &
       ia2=40692,iq1=53668,iq2=52774,ir1=12211, &
       ir2=3791,ntab=32,ndiv=1+imm1/ntab, &
       eps=1.2d-7,rnmx=1.0d0-eps)

      INTEGER idum2,j,k,iv(Ntab),iy

      SAVE iv,iy,idum2
      DATA idum2/123456789/, iv/ntab*0/, iy/0/

      IF (idum.LE.0) THEN
        idum=MAX(-idum,1)
        idum2=idum
        DO j=ntab+8,1,-1
          k=idum/iq1
          idum=ia1*(idum-k*iq1)-k*ir1
          IF (idum.LT.0) idum=idum+im1
          IF (j.LE.ntab) iv(j)=idum
        ENDDO
        iy=iv(1)
      ENDIF

      k=idum/iq1
      idum=ia1*(idum-k*iq1)-k*ir1
      IF (idum.LT.0) idum=idum+im1
      k=idum2/iq2
      idum2=ia2*(idum2-k*iq2)-k*ir2
      IF (idum2.LT.0) idum2=idum2+im2
      j=1+iy/ndiv
      iy=iv(j)-idum2
      iv(j)=idum
      IF(iy.LT.1)iy=iy+imm1
      ran2=Min(am*iy,rnmx)

      RETURN
      END


      FUNCTION rangauss()

! Purpose:
! To generate random numbers from a gaussian distribution.
```

```
    DOUBLE PRECISION ran_uniform, rangauss, v1, v2, rsq

100 v1=2.0D0*ran_uniform()-1.0D0
    v2=2.0D0*ran_uniform()-1.0D0

    rsq=v1*v1+v2*v2

    IF (rsq .GE. 1.0D0 .OR. rsq .LE. 0.0D0) GOTO 100

    rangauss=v1*DSQRT(-2.0D0*DLOG(rsq)/rsq)

    RETURN
    END
```

3. THE INITIAL STRUCTURE FILE

```
2.161D1  2.161D1  2.161D1  ! Box size (angstroms) 2x2x2 FCC cells
! Type  x         y         z       mass      charge
La    5.4025         0         0  138.9055    3.0000
La    8.1037    2.7012         0  138.9055    3.0000
La    8.1037         0    2.7012  138.9055    3.0000
La    5.4025    2.7012    2.7012  138.9055    3.0000
La         0    5.4025         0  138.9055    3.0000
La    2.7012    8.1037         0  138.9055    3.0000
La    2.7012    5.4025    2.7012  138.9055    3.0000
La         0    8.1037    2.7012  138.9055    3.0000
La         0         0    5.4025  138.9055    3.0000
La    2.7012    2.7012    5.4025  138.9055    3.0000
La    2.7012         0    8.1037  138.9055    3.0000
La         0    2.7012    8.1037  138.9055    3.0000
La    5.4025    5.4025    5.4025  138.9055    3.0000
La    8.1037    8.1037    5.4025  138.9055    3.0000
La    8.1037    5.4025    8.1037  138.9055    3.0000
La    5.4025    8.1037    8.1037  138.9055    3.0000
La    5.4025         0   10.8050  138.9055    3.0000
La    8.1037    2.7012   10.8050  138.9055    3.0000
La    8.1037         0   13.5063  138.9055    3.0000
La    5.4025    2.7012   13.5063  138.9055    3.0000
La         0    5.4025   10.8050  138.9055    3.0000
La    2.7012    8.1037   10.8050  138.9055    3.0000
La    2.7012    5.4025   13.5063  138.9055    3.0000
La         0    8.1037   13.5063  138.9055    3.0000
La         0         0   16.2075  138.9055    3.0000
La    2.7012    2.7012   16.2075  138.9055    3.0000
La    2.7012         0   18.9087  138.9055    3.0000
La         0    2.7012   18.9087  138.9055    3.0000
La    5.4025    5.4025   16.2075  138.9055    3.0000
La    8.1037    8.1037   16.2075  138.9055    3.0000
La    8.1037    5.4025   18.9087  138.9055    3.0000
La    5.4025    8.1037   18.9087  138.9055    3.0000
La    5.4025   10.8050         0  138.9055    3.0000
La    8.1037   13.5063         0  138.9055    3.0000
La    8.1037   10.8050    2.7012  138.9055    3.0000
```

```
La    5.4025   13.5063    2.7012  138.9055    3.0000
La         0   16.2075         0  138.9055    3.0000
La    2.7012   18.9087         0  138.9055    3.0000
La    2.7012   16.2075    2.7012  138.9055    3.0000
La         0   18.9087    2.7012  138.9055    3.0000
La         0   10.8050    5.4025  138.9055    3.0000
La    2.7012   13.5063    5.4025  138.9055    3.0000
La    2.7012   10.8050    8.1037  138.9055    3.0000
La         0   13.5063    8.1037  138.9055    3.0000
La    5.4025   16.2075    5.4025  138.9055    3.0000
La    8.1037   18.9087    5.4025  138.9055    3.0000
La    8.1037   16.2075    8.1037  138.9055    3.0000
La    5.4025   18.9087    8.1037  138.9055    3.0000
La    5.4025   10.8050   10.8050  138.9055    3.0000
La    8.1037   13.5063   10.8050  138.9055    3.0000
La    8.1037   10.8050   13.5063  138.9055    3.0000
La    5.4025   13.5063   13.5063  138.9055    3.0000
La         0   16.2075   10.8050  138.9055    3.0000
La    2.7012   18.9087   10.8050  138.9055    3.0000
La    2.7012   16.2075   13.5063  138.9055    3.0000
La         0   18.9087   13.5063  138.9055    3.0000
La         0   10.8050   16.2075  138.9055    3.0000
La    2.7012   13.5063   16.2075  138.9055    3.0000
La    2.7012   10.8050   18.9087  138.9055    3.0000
La         0   13.5063   18.9087  138.9055    3.0000
La    5.4025   16.2075   16.2075  138.9055    3.0000
La    8.1037   18.9087   16.2075  138.9055    3.0000
La    8.1037   16.2075   18.9087  138.9055    3.0000
La    5.4025   18.9087   18.9087  138.9055    3.0000
La   16.2075         0         0  138.9055    3.0000
La   18.9087    2.7012         0  138.9055    3.0000
La   18.9087         0    2.7012  138.9055    3.0000
La   16.2075    2.7012    2.7012  138.9055    3.0000
La   10.8050    5.4025         0  138.9055    3.0000
La   13.5063    8.1037         0  138.9055    3.0000
La   13.5063    5.4025    2.7012  138.9055    3.0000
La   10.8050    8.1037    2.7012  138.9055    3.0000
La   10.8050         0    5.4025  138.9055    3.0000
La   13.5063    2.7012    5.4025  138.9055    3.0000
La   13.5063         0    8.1037  138.9055    3.0000
La   10.8050    2.7012    8.1037  138.9055    3.0000
La   16.2075    5.4025    5.4025  138.9055    3.0000
La   18.9087    8.1037    5.4025  138.9055    3.0000
La   18.9087    5.4025    8.1037  138.9055    3.0000
La   16.2075    8.1037    8.1037  138.9055    3.0000
La   16.2075         0   10.8050  138.9055    3.0000
La   18.9087    2.7012   10.8050  138.9055    3.0000
La   18.9087         0   13.5063  138.9055    3.0000
La   16.2075    2.7012   13.5063  138.9055    3.0000
La   10.8050    5.4025   10.8050  138.9055    3.0000
La   13.5063    8.1037   10.8050  138.9055    3.0000
La   13.5063    5.4025   13.5063  138.9055    3.0000
La   10.8050    8.1037   13.5063  138.9055    3.0000
La   10.8050         0   16.2075  138.9055    3.0000
La   13.5063    2.7012   16.2075  138.9055    3.0000
```

```
La   13.5063         0   18.9087  138.9055    3.0000
La   10.8050    2.7012   18.9087  138.9055    3.0000
La   16.2075    5.4025   16.2075  138.9055    3.0000
La   18.9087    8.1037   16.2075  138.9055    3.0000
La   18.9087    5.4025   18.9087  138.9055    3.0000
La   16.2075    8.1037   18.9087  138.9055    3.0000
La   16.2075   10.8050         0  138.9055    3.0000
La   18.9087   13.5063         0  138.9055    3.0000
La   18.9087   10.8050    2.7012  138.9055    3.0000
La   16.2075   13.5063    2.7012  138.9055    3.0000
La   10.8050   16.2075         0  138.9055    3.0000
La   13.5063   18.9087         0  138.9055    3.0000
La   13.5063   16.2075    2.7012  138.9055    3.0000
La   10.8050   18.9087    2.7012  138.9055    3.0000
La   10.8050   10.8050    5.4025  138.9055    3.0000
La   13.5063   13.5063    5.4025  138.9055    3.0000
La   13.5063   10.8050    8.1037  138.9055    3.0000
La   10.8050   13.5063    8.1037  138.9055    3.0000
La   16.2075   16.2075    5.4025  138.9055    3.0000
La   18.9087   18.9087    5.4025  138.9055    3.0000
La   18.9087   16.2075    8.1037  138.9055    3.0000
La   16.2075   18.9087    8.1037  138.9055    3.0000
La   16.2075   10.8050   10.8050  138.9055    3.0000
La   18.9087   13.5063   10.8050  138.9055    3.0000
La   18.9087   10.8050   13.5063  138.9055    3.0000
La   16.2075   13.5063   13.5063  138.9055    3.0000
La   10.8050   16.2075   10.8050  138.9055    3.0000
La   13.5063   18.9087   10.8050  138.9055    3.0000
La   13.5063   16.2075   13.5063  138.9055    3.0000
La   10.8050   18.9087   13.5063  138.9055    3.0000
La   10.8050   10.8050   16.2075  138.9055    3.0000
La   13.5063   13.5063   16.2075  138.9055    3.0000
La   13.5063   10.8050   18.9087  138.9055    3.0000
La   10.8050   13.5063   18.9087  138.9055    3.0000
La   16.2075   16.2075   16.2075  138.9055    3.0000
La   18.9087   18.9087   16.2075  138.9055    3.0000
La   18.9087   16.2075   18.9087  138.9055    3.0000
La   16.2075   18.9087   18.9087  138.9055    3.0000
Zr         0         0         0   91.2240    4.0000
Zr    2.7012    2.7012         0   91.2240    4.0000
Zr    2.7012         0    2.7012   91.2240    4.0000
Zr         0    2.7012    2.7012   91.2240    4.0000
Zr    5.4025    5.4025         0   91.2240    4.0000
Zr    8.1037    8.1037         0   91.2240    4.0000
Zr    8.1037    5.4025    2.7012   91.2240    4.0000
Zr    5.4025    8.1037    2.7012   91.2240    4.0000
Zr    5.4025         0    5.4025   91.2240    4.0000
Zr    8.1037    2.7012    5.4025   91.2240    4.0000
Zr    8.1037         0    8.1037   91.2240    4.0000
Zr    5.4025    2.7012    8.1037   91.2240    4.0000
Zr         0    5.4025    5.4025   91.2240    4.0000
Zr    2.7012    8.1037    5.4025   91.2240    4.0000
Zr    2.7012    5.4025    8.1037   91.2240    4.0000
Zr         0    8.1037    8.1037   91.2240    4.0000
Zr         0         0   10.8050   91.2240    4.0000
```

Zr	2.7012	2.7012	10.8050	91.2240	4.0000
Zr	2.7012	0	13.5063	91.2240	4.0000
Zr	0	2.7012	13.5063	91.2240	4.0000
Zr	5.4025	5.4025	10.8050	91.2240	4.0000
Zr	8.1037	8.1037	10.8050	91.2240	4.0000
Zr	8.1037	5.4025	13.5063	91.2240	4.0000
Zr	5.4025	8.1037	13.5063	91.2240	4.0000
Zr	5.4025	0	16.2075	91.2240	4.0000
Zr	8.1037	2.7012	16.2075	91.2240	4.0000
Zr	8.1037	0	18.9087	91.2240	4.0000
Zr	5.4025	2.7012	18.9087	91.2240	4.0000
Zr	0	5.4025	16.2075	91.2240	4.0000
Zr	2.7012	8.1037	16.2075	91.2240	4.0000
Zr	2.7012	5.4025	18.9087	91.2240	4.0000
Zr	0	8.1037	18.9087	91.2240	4.0000
Zr	0	10.8050	0	91.2240	4.0000
Zr	2.7012	13.5063	0	91.2240	4.0000
Zr	2.7012	10.8050	2.7012	91.2240	4.0000
Zr	0	13.5063	2.7012	91.2240	4.0000
Zr	5.4025	16.2075	0	91.2240	4.0000
Zr	8.1037	18.9087	0	91.2240	4.0000
Zr	8.1037	16.2075	2.7012	91.2240	4.0000
Zr	5.4025	18.9087	2.7012	91.2240	4.0000
Zr	5.4025	10.8050	5.4025	91.2240	4.0000
Zr	8.1037	13.5063	5.4025	91.2240	4.0000
Zr	8.1037	10.8050	8.1037	91.2240	4.0000
Zr	5.4025	13.5063	8.1037	91.2240	4.0000
Zr	0	16.2075	5.4025	91.2240	4.0000
Zr	2.7012	18.9087	5.4025	91.2240	4.0000
Zr	2.7012	16.2075	8.1037	91.2240	4.0000
Zr	0	18.9087	8.1037	91.2240	4.0000
Zr	0	10.8050	10.8050	91.2240	4.0000
Zr	2.7012	13.5063	10.8050	91.2240	4.0000
Zr	2.7012	10.8050	13.5063	91.2240	4.0000
Zr	0	13.5063	13.5063	91.2240	4.0000
Zr	5.4025	16.2075	10.8050	91.2240	4.0000
Zr	8.1037	18.9087	10.8050	91.2240	4.0000
Zr	8.1037	16.2075	13.5063	91.2240	4.0000
Zr	5.4025	18.9087	13.5063	91.2240	4.0000
Zr	5.4025	10.8050	16.2075	91.2240	4.0000
Zr	8.1037	13.5063	16.2075	91.2240	4.0000
Zr	8.1037	10.8050	18.9087	91.2240	4.0000
Zr	5.4025	13.5063	18.9087	91.2240	4.0000
Zr	0	16.2075	16.2075	91.2240	4.0000
Zr	2.7012	18.9087	16.2075	91.2240	4.0000
Zr	2.7012	16.2075	18.9087	91.2240	4.0000
Zr	0	18.9087	18.9087	91.2240	4.0000
Zr	10.8050	0	0	91.2240	4.0000
Zr	13.5063	2.7012	0	91.2240	4.0000
Zr	13.5063	0	2.7012	91.2240	4.0000
Zr	10.8050	2.7012	2.7012	91.2240	4.0000
Zr	16.2075	5.4025	0	91.2240	4.0000
Zr	18.9087	8.1037	0	91.2240	4.0000
Zr	18.9087	5.4025	2.7012	91.2240	4.0000
Zr	16.2075	8.1037	2.7012	91.2240	4.0000

```
Zr   16.2075         0    5.4025   91.2240    4.0000
Zr   18.9087    2.7012    5.4025   91.2240    4.0000
Zr   18.9087         0    8.1037   91.2240    4.0000
Zr   16.2075    2.7012    8.1037   91.2240    4.0000
Zr   10.8050    5.4025    5.4025   91.2240    4.0000
Zr   13.5063    8.1037    5.4025   91.2240    4.0000
Zr   13.5063    5.4025    8.1037   91.2240    4.0000
Zr   10.8050    8.1037    8.1037   91.2240    4.0000
Zr   10.8050         0   10.8050   91.2240    4.0000
Zr   13.5063    2.7012   10.8050   91.2240    4.0000
Zr   13.5063         0   13.5063   91.2240    4.0000
Zr   10.8050    2.7012   13.5063   91.2240    4.0000
Zr   16.2075    5.4025   10.8050   91.2240    4.0000
Zr   18.9087    8.1037   10.8050   91.2240    4.0000
Zr   18.9087    5.4025   13.5063   91.2240    4.0000
Zr   16.2075    8.1037   13.5063   91.2240    4.0000
Zr   16.2075         0   16.2075   91.2240    4.0000
Zr   18.9087    2.7012   16.2075   91.2240    4.0000
Zr   18.9087         0   18.9087   91.2240    4.0000
Zr   16.2075    2.7012   18.9087   91.2240    4.0000
Zr   10.8050    5.4025   16.2075   91.2240    4.0000
Zr   13.5063    8.1037   16.2075   91.2240    4.0000
Zr   13.5063    5.4025   18.9087   91.2240    4.0000
Zr   10.8050    8.1037   18.9087   91.2240    4.0000
Zr   10.8050   10.8050         0   91.2240    4.0000
Zr   13.5063   13.5063         0   91.2240    4.0000
Zr   13.5063   10.8050    2.7012   91.2240    4.0000
Zr   10.8050   13.5063    2.7012   91.2240    4.0000
Zr   16.2075   16.2075         0   91.2240    4.0000
Zr   18.9087   18.9087         0   91.2240    4.0000
Zr   18.9087   16.2075    2.7012   91.2240    4.0000
Zr   16.2075   18.9087    2.7012   91.2240    4.0000
Zr   16.2075   10.8050    5.4025   91.2240    4.0000
Zr   18.9087   13.5063    5.4025   91.2240    4.0000
Zr   18.9087   10.8050    8.1037   91.2240    4.0000
Zr   16.2075   13.5063    8.1037   91.2240    4.0000
Zr   10.8050   16.2075    5.4025   91.2240    4.0000
Zr   13.5063   18.9087    5.4025   91.2240    4.0000
Zr   13.5063   16.2075    8.1037   91.2240    4.0000
Zr   10.8050   18.9087    8.1037   91.2240    4.0000
Zr   10.8050   10.8050   10.8050   91.2240    4.0000
Zr   13.5063   13.5063   10.8050   91.2240    4.0000
Zr   13.5063   10.8050   13.5063   91.2240    4.0000
Zr   10.8050   13.5063   13.5063   91.2240    4.0000
Zr   16.2075   16.2075   10.8050   91.2240    4.0000
Zr   18.9087   18.9087   10.8050   91.2240    4.0000
Zr   18.9087   16.2075   13.5063   91.2240    4.0000
Zr   16.2075   18.9087   13.5063   91.2240    4.0000
Zr   16.2075   10.8050   16.2075   91.2240    4.0000
Zr   18.9087   13.5063   16.2075   91.2240    4.0000
Zr   18.9087   10.8050   18.9087   91.2240    4.0000
Zr   16.2075   13.5063   18.9087   91.2240    4.0000
Zr   10.8050   16.2075   16.2075   91.2240    4.0000
Zr   13.5063   18.9087   16.2075   91.2240    4.0000
Zr   13.5063   16.2075   18.9087   91.2240    4.0000
```

Zr	10.8050	18.9087	18.9087	91.2240	4.0000
O	4.5381	1.3506	1.3506	15.9994	-2.0000
O	1.3506	4.5381	1.3506	15.9994	-2.0000
O	4.0519	4.0519	1.8369	15.9994	-2.0000
O	1.3506	1.3506	4.5381	15.9994	-2.0000
O	4.0519	1.8369	4.0519	15.9994	-2.0000
O	1.8369	4.0519	4.0519	15.9994	-2.0000
O	4.0519	4.0519	4.0519	15.9994	-2.0000
O	9.9406	1.3506	1.3506	15.9994	-2.0000
O	6.7531	4.5381	1.3506	15.9994	-2.0000
O	9.4544	4.0519	1.8369	15.9994	-2.0000
O	6.7531	1.3506	4.5381	15.9994	-2.0000
O	9.4544	1.8369	4.0519	15.9994	-2.0000
O	7.2394	4.0519	4.0519	15.9994	-2.0000
O	9.4544	4.0519	4.0519	15.9994	-2.0000
O	9.9406	6.7531	1.3506	15.9994	-2.0000
O	6.7531	9.9406	1.3506	15.9994	-2.0000
O	9.4544	9.4544	1.8369	15.9994	-2.0000
O	6.7531	6.7531	4.5381	15.9994	-2.0000
O	9.4544	7.2394	4.0519	15.9994	-2.0000
O	7.2394	9.4544	4.0519	15.9994	-2.0000
O	9.4544	9.4544	4.0519	15.9994	-2.0000
O	4.5381	6.7531	1.3506	15.9994	-2.0000
O	1.3506	9.9406	1.3506	15.9994	-2.0000
O	4.0519	9.4544	1.8369	15.9994	-2.0000
O	1.3506	6.7531	4.5381	15.9994	-2.0000
O	4.0519	7.2394	4.0519	15.9994	-2.0000
O	1.8369	9.4544	4.0519	15.9994	-2.0000
O	4.0519	9.4544	4.0519	15.9994	-2.0000
O	8.9682	1.3506	6.7531	15.9994	-2.0000
O	9.4544	4.0519	6.2669	15.9994	-2.0000
O	6.7531	3.5657	6.7531	15.9994	-2.0000
O	6.7531	1.3506	8.9682	15.9994	-2.0000
O	9.4544	0.8644	9.4544	15.9994	-2.0000
O	6.2669	4.0519	9.4544	15.9994	-2.0000
O	6.7531	1.3506	6.7531	15.9994	-2.0000
O	3.5657	1.3506	6.7531	15.9994	-2.0000
O	4.0519	4.0519	6.2669	15.9994	-2.0000
O	1.3506	3.5657	6.7531	15.9994	-2.0000
O	1.3506	1.3506	8.9682	15.9994	-2.0000
O	4.0519	0.8644	9.4544	15.9994	-2.0000
O	0.8644	4.0519	9.4544	15.9994	-2.0000
O	1.3506	1.3506	6.7531	15.9994	-2.0000
O	3.5657	6.7531	6.7531	15.9994	-2.0000
O	4.0519	9.4544	6.2669	15.9994	-2.0000
O	1.3506	8.9682	6.7531	15.9994	-2.0000
O	1.3506	6.7531	8.9682	15.9994	-2.0000
O	4.0519	6.2669	9.4544	15.9994	-2.0000
O	0.8644	9.4544	9.4544	15.9994	-2.0000
O	1.3506	6.7531	6.7531	15.9994	-2.0000
O	8.9682	6.7531	6.7531	15.9994	-2.0000
O	9.4544	9.4544	6.2669	15.9994	-2.0000
O	6.7531	8.9682	6.7531	15.9994	-2.0000
O	6.7531	6.7531	8.9682	15.9994	-2.0000
O	9.4544	6.2669	9.4544	15.9994	-2.0000

O	6.2669	9.4544	9.4544	15.9994	-2.0000
O	6.7531	6.7531	6.7531	15.9994	-2.0000
O	4.5381	1.3506	12.1556	15.9994	-2.0000
O	1.3506	4.5381	12.1556	15.9994	-2.0000
O	4.0519	4.0519	12.6419	15.9994	-2.0000
O	1.3506	1.3506	15.3431	15.9994	-2.0000
O	4.0519	1.8369	14.8569	15.9994	-2.0000
O	1.8369	4.0519	14.8569	15.9994	-2.0000
O	4.0519	4.0519	14.8569	15.9994	-2.0000
O	9.9406	1.3506	12.1556	15.9994	-2.0000
O	6.7531	4.5381	12.1556	15.9994	-2.0000
O	9.4544	4.0519	12.6419	15.9994	-2.0000
O	6.7531	1.3506	15.3431	15.9994	-2.0000
O	9.4544	1.8369	14.8569	15.9994	-2.0000
O	7.2394	4.0519	14.8569	15.9994	-2.0000
O	9.4544	4.0519	14.8569	15.9994	-2.0000
O	9.9406	6.7531	12.1556	15.9994	-2.0000
O	6.7531	9.9406	12.1556	15.9994	-2.0000
O	9.4544	9.4544	12.6419	15.9994	-2.0000
O	6.7531	6.7531	15.3431	15.9994	-2.0000
O	9.4544	7.2394	14.8569	15.9994	-2.0000
O	7.2394	9.4544	14.8569	15.9994	-2.0000
O	9.4544	9.4544	14.8569	15.9994	-2.0000
O	4.5381	6.7531	12.1556	15.9994	-2.0000
O	1.3506	9.9406	12.1556	15.9994	-2.0000
O	4.0519	9.4544	12.6419	15.9994	-2.0000
O	1.3506	6.7531	15.3431	15.9994	-2.0000
O	4.0519	7.2394	14.8569	15.9994	-2.0000
O	1.8369	9.4544	14.8569	15.9994	-2.0000
O	4.0519	9.4544	14.8569	15.9994	-2.0000
O	8.9682	1.3506	17.5581	15.9994	-2.0000
O	9.4544	4.0519	17.0719	15.9994	-2.0000
O	6.7531	3.5657	17.5581	15.9994	-2.0000
O	6.7531	1.3506	19.7732	15.9994	-2.0000
O	9.4544	0.8644	20.2594	15.9994	-2.0000
O	6.2669	4.0519	20.2594	15.9994	-2.0000
O	6.7531	1.3506	17.5581	15.9994	-2.0000
O	3.5657	1.3506	17.5581	15.9994	-2.0000
O	4.0519	4.0519	17.0719	15.9994	-2.0000
O	1.3506	3.5657	17.5581	15.9994	-2.0000
O	1.3506	1.3506	19.7732	15.9994	-2.0000
O	4.0519	0.8644	20.2594	15.9994	-2.0000
O	0.8644	4.0519	20.2594	15.9994	-2.0000
O	1.3506	1.3506	17.5581	15.9994	-2.0000
O	3.5657	6.7531	17.5581	15.9994	-2.0000
O	4.0519	9.4544	17.0719	15.9994	-2.0000
O	1.3506	8.9682	17.5581	15.9994	-2.0000
O	1.3506	6.7531	19.7732	15.9994	-2.0000
O	4.0519	6.2669	20.2594	15.9994	-2.0000
O	0.8644	9.4544	20.2594	15.9994	-2.0000
O	1.3506	6.7531	17.5581	15.9994	-2.0000
O	8.9682	6.7531	17.5581	15.9994	-2.0000
O	9.4544	9.4544	17.0719	15.9994	-2.0000
O	6.7531	8.9682	17.5581	15.9994	-2.0000
O	6.7531	6.7531	19.7732	15.9994	-2.0000

```
O     9.4544    6.2669   20.2594   15.9994   -2.0000
O     6.2669    9.4544   20.2594   15.9994   -2.0000
O     6.7531    6.7531   17.5581   15.9994   -2.0000
O     4.5381   12.1556    1.3506   15.9994   -2.0000
O     1.3506   15.3431    1.3506   15.9994   -2.0000
O     4.0519   14.8569    1.8369   15.9994   -2.0000
O     1.3506   12.1556    4.5381   15.9994   -2.0000
O     4.0519   12.6419    4.0519   15.9994   -2.0000
O     1.8369   14.8569    4.0519   15.9994   -2.0000
O     4.0519   14.8569    4.0519   15.9994   -2.0000
O     9.9406   12.1556    1.3506   15.9994   -2.0000
O     6.7531   15.3431    1.3506   15.9994   -2.0000
O     9.4544   14.8569    1.8369   15.9994   -2.0000
O     6.7531   12.1556    4.5381   15.9994   -2.0000
O     9.4544   12.6419    4.0519   15.9994   -2.0000
O     7.2394   14.8569    4.0519   15.9994   -2.0000
O     9.4544   14.8569    4.0519   15.9994   -2.0000
O     9.9406   17.5581    1.3506   15.9994   -2.0000
O     6.7531   20.7456    1.3506   15.9994   -2.0000
O     9.4544   20.2594    1.8369   15.9994   -2.0000
O     6.7531   17.5581    4.5381   15.9994   -2.0000
O     9.4544   18.0444    4.0519   15.9994   -2.0000
O     7.2394   20.2594    4.0519   15.9994   -2.0000
O     9.4544   20.2594    4.0519   15.9994   -2.0000
O     4.5381   17.5581    1.3506   15.9994   -2.0000
O     1.3506   20.7456    1.3506   15.9994   -2.0000
O     4.0519   20.2594    1.8369   15.9994   -2.0000
O     1.3506   17.5581    4.5381   15.9994   -2.0000
O     4.0519   18.0444    4.0519   15.9994   -2.0000
O     1.8369   20.2594    4.0519   15.9994   -2.0000
O     4.0519   20.2594    4.0519   15.9994   -2.0000
O     8.9682   12.1556    6.7531   15.9994   -2.0000
O     9.4544   14.8569    6.2669   15.9994   -2.0000
O     6.7531   14.3706    6.7531   15.9994   -2.0000
O     6.7531   12.1556    8.9682   15.9994   -2.0000
O     9.4544   11.6694    9.4544   15.9994   -2.0000
O     6.2669   14.8569    9.4544   15.9994   -2.0000
O     6.7531   12.1556    6.7531   15.9994   -2.0000
O     3.5657   12.1556    6.7531   15.9994   -2.0000
O     4.0519   14.8569    6.2669   15.9994   -2.0000
O     1.3506   14.3706    6.7531   15.9994   -2.0000
O     1.3506   12.1556    8.9682   15.9994   -2.0000
O     4.0519   11.6694    9.4544   15.9994   -2.0000
O     0.8644   14.8569    9.4544   15.9994   -2.0000
O     1.3506   12.1556    6.7531   15.9994   -2.0000
O     3.5657   17.5581    6.7531   15.9994   -2.0000
O     4.0519   20.2594    6.2669   15.9994   -2.0000
O     1.3506   19.7732    6.7531   15.9994   -2.0000
O     1.3506   17.5581    8.9682   15.9994   -2.0000
O     4.0519   17.0719    9.4544   15.9994   -2.0000
O     0.8644   20.2594    9.4544   15.9994   -2.0000
O     1.3506   17.5581    6.7531   15.9994   -2.0000
O     8.9682   17.5581    6.7531   15.9994   -2.0000
O     9.4544   20.2594    6.2669   15.9994   -2.0000
O     6.7531   19.7732    6.7531   15.9994   -2.0000
```

O	6.7531	17.5581	8.9682	15.9994	-2.0000
O	9.4544	17.0719	9.4544	15.9994	-2.0000
O	6.2669	20.2594	9.4544	15.9994	-2.0000
O	6.7531	17.5581	6.7531	15.9994	-2.0000
O	4.5381	12.1556	12.1556	15.9994	-2.0000
O	1.3506	15.3431	12.1556	15.9994	-2.0000
O	4.0519	14.8569	12.6419	15.9994	-2.0000
O	1.3506	12.1556	15.3431	15.9994	-2.0000
O	4.0519	12.6419	14.8569	15.9994	-2.0000
O	1.8369	14.8569	14.8569	15.9994	-2.0000
O	4.0519	14.8569	14.8569	15.9994	-2.0000
O	9.9406	12.1556	12.1556	15.9994	-2.0000
O	6.7531	15.3431	12.1556	15.9994	-2.0000
O	9.4544	14.8569	12.6419	15.9994	-2.0000
O	6.7531	12.1556	15.3431	15.9994	-2.0000
O	9.4544	12.6419	14.8569	15.9994	-2.0000
O	7.2394	14.8569	14.8569	15.9994	-2.0000
O	9.4544	14.8569	14.8569	15.9994	-2.0000
O	9.9406	17.5581	12.1556	15.9994	-2.0000
O	6.7531	20.7456	12.1556	15.9994	-2.0000
O	9.4544	20.2594	12.6419	15.9994	-2.0000
O	6.7531	17.5581	15.3431	15.9994	-2.0000
O	9.4544	18.0444	14.8569	15.9994	-2.0000
O	7.2394	20.2594	14.8569	15.9994	-2.0000
O	9.4544	20.2594	14.8569	15.9994	-2.0000
O	4.5381	17.5581	12.1556	15.9994	-2.0000
O	1.3506	20.7456	12.1556	15.9994	-2.0000
O	4.0519	20.2594	12.6419	15.9994	-2.0000
O	1.3506	17.5581	15.3431	15.9994	-2.0000
O	4.0519	18.0444	14.8569	15.9994	-2.0000
O	1.8369	20.2594	14.8569	15.9994	-2.0000
O	4.0519	20.2594	14.8569	15.9994	-2.0000
O	8.9682	12.1556	17.5581	15.9994	-2.0000
O	9.4544	14.8569	17.0719	15.9994	-2.0000
O	6.7531	14.3706	17.5581	15.9994	-2.0000
O	6.7531	12.1556	19.7732	15.9994	-2.0000
O	9.4544	11.6694	20.2594	15.9994	-2.0000
O	6.2669	14.8569	20.2594	15.9994	-2.0000
O	6.7531	12.1556	17.5581	15.9994	-2.0000
O	3.5657	12.1556	17.5581	15.9994	-2.0000
O	4.0519	14.8569	17.0719	15.9994	-2.0000
O	1.3506	14.3706	17.5581	15.9994	-2.0000
O	1.3506	12.1556	19.7732	15.9994	-2.0000
O	4.0519	11.6694	20.2594	15.9994	-2.0000
O	0.8644	14.8569	20.2594	15.9994	-2.0000
O	1.3506	12.1556	17.5581	15.9994	-2.0000
O	3.5657	17.5581	17.5581	15.9994	-2.0000
O	4.0519	20.2594	17.0719	15.9994	-2.0000
O	1.3506	19.7732	17.5581	15.9994	-2.0000
O	1.3506	17.5581	19.7732	15.9994	-2.0000
O	4.0519	17.0719	20.2594	15.9994	-2.0000
O	0.8644	20.2594	20.2594	15.9994	-2.0000
O	1.3506	17.5581	17.5581	15.9994	-2.0000
O	8.9682	17.5581	17.5581	15.9994	-2.0000
O	9.4544	20.2594	17.0719	15.9994	-2.0000

```
O     6.7531   19.7732   17.5581   15.9994   -2.0000
O     6.7531   17.5581   19.7732   15.9994   -2.0000
O     9.4544   17.0719   20.2594   15.9994   -2.0000
O     6.2669   20.2594   20.2594   15.9994   -2.0000
O     6.7531   17.5581   17.5581   15.9994   -2.0000
O    15.3431    1.3506    1.3506   15.9994   -2.0000
O    12.1556    4.5381    1.3506   15.9994   -2.0000
O    14.8569    4.0519    1.8369   15.9994   -2.0000
O    12.1556    1.3506    4.5381   15.9994   -2.0000
O    14.8569    1.8369    4.0519   15.9994   -2.0000
O    12.6419    4.0519    4.0519   15.9994   -2.0000
O    14.8569    4.0519    4.0519   15.9994   -2.0000
O    20.7456    1.3506    1.3506   15.9994   -2.0000
O    17.5581    4.5381    1.3506   15.9994   -2.0000
O    20.2594    4.0519    1.8369   15.9994   -2.0000
O    17.5581    1.3506    4.5381   15.9994   -2.0000
O    20.2594    1.8369    4.0519   15.9994   -2.0000
O    18.0444    4.0519    4.0519   15.9994   -2.0000
O    20.2594    4.0519    4.0519   15.9994   -2.0000
O    20.7456    6.7531    1.3506   15.9994   -2.0000
O    17.5581    9.9406    1.3506   15.9994   -2.0000
O    20.2594    9.4544    1.8369   15.9994   -2.0000
O    17.5581    6.7531    4.5381   15.9994   -2.0000
O    20.2594    7.2394    4.0519   15.9994   -2.0000
O    18.0444    9.4544    4.0519   15.9994   -2.0000
O    20.2594    9.4544    4.0519   15.9994   -2.0000
O    15.3431    6.7531    1.3506   15.9994   -2.0000
O    12.1556    9.9406    1.3506   15.9994   -2.0000
O    14.8569    9.4544    1.8369   15.9994   -2.0000
O    12.1556    6.7531    4.5381   15.9994   -2.0000
O    14.8569    7.2394    4.0519   15.9994   -2.0000
O    12.6419    9.4544    4.0519   15.9994   -2.0000
O    14.8569    9.4544    4.0519   15.9994   -2.0000
O    19.7732    1.3506    6.7531   15.9994   -2.0000
O    20.2594    4.0519    6.2669   15.9994   -2.0000
O    17.5581    3.5657    6.7531   15.9994   -2.0000
O    17.5581    1.3506    8.9682   15.9994   -2.0000
O    20.2594    0.8644    9.4544   15.9994   -2.0000
O    17.0719    4.0519    9.4544   15.9994   -2.0000
O    17.5581    1.3506    6.7531   15.9994   -2.0000
O    14.3706    1.3506    6.7531   15.9994   -2.0000
O    14.8569    4.0519    6.2669   15.9994   -2.0000
O    12.1556    3.5657    6.7531   15.9994   -2.0000
O    12.1556    1.3506    8.9682   15.9994   -2.0000
O    14.8569    0.8644    9.4544   15.9994   -2.0000
O    11.6694    4.0519    9.4544   15.9994   -2.0000
O    12.1556    1.3506    6.7531   15.9994   -2.0000
O    14.3706    6.7531    6.7531   15.9994   -2.0000
O    14.8569    9.4544    6.2669   15.9994   -2.0000
O    12.1556    8.9682    6.7531   15.9994   -2.0000
O    12.1556    6.7531    8.9682   15.9994   -2.0000
O    14.8569    6.2669    9.4544   15.9994   -2.0000
O    11.6694    9.4544    9.4544   15.9994   -2.0000
O    12.1556    6.7531    6.7531   15.9994   -2.0000
O    19.7732    6.7531    6.7531   15.9994   -2.0000
```

```
O    20.2594    9.4544    6.2669   15.9994   -2.0000
O    17.5581    8.9682    6.7531   15.9994   -2.0000
O    17.5581    6.7531    8.9682   15.9994   -2.0000
O    20.2594    6.2669    9.4544   15.9994   -2.0000
O    17.0719    9.4544    9.4544   15.9994   -2.0000
O    17.5581    6.7531    6.7531   15.9994   -2.0000
O    15.3431    1.3506   12.1556   15.9994   -2.0000
O    12.1556    4.5381   12.1556   15.9994   -2.0000
O    14.8569    4.0519   12.6419   15.9994   -2.0000
O    12.1556    1.3506   15.3431   15.9994   -2.0000
O    14.8569    1.8369   14.8569   15.9994   -2.0000
O    12.6419    4.0519   14.8569   15.9994   -2.0000
O    14.8569    4.0519   14.8569   15.9994   -2.0000
O    20.7456    1.3506   12.1556   15.9994   -2.0000
O    17.5581    4.5381   12.1556   15.9994   -2.0000
O    20.2594    4.0519   12.6419   15.9994   -2.0000
O    17.5581    1.3506   15.3431   15.9994   -2.0000
O    20.2594    1.8369   14.8569   15.9994   -2.0000
O    18.0444    4.0519   14.8569   15.9994   -2.0000
O    20.2594    4.0519   14.8569   15.9994   -2.0000
O    20.7456    6.7531   12.1556   15.9994   -2.0000
O    17.5581    9.9406   12.1556   15.9994   -2.0000
O    20.2594    9.4544   12.6419   15.9994   -2.0000
O    17.5581    6.7531   15.3431   15.9994   -2.0000
O    20.2594    7.2394   14.8569   15.9994   -2.0000
O    18.0444    9.4544   14.8569   15.9994   -2.0000
O    20.2594    9.4544   14.8569   15.9994   -2.0000
O    15.3431    6.7531   12.1556   15.9994   -2.0000
O    12.1556    9.9406   12.1556   15.9994   -2.0000
O    14.8569    9.4544   12.6419   15.9994   -2.0000
O    12.1556    6.7531   15.3431   15.9994   -2.0000
O    14.8569    7.2394   14.8569   15.9994   -2.0000
O    12.6419    9.4544   14.8569   15.9994   -2.0000
O    14.8569    9.4544   14.8569   15.9994   -2.0000
O    19.7732    1.3506   17.5581   15.9994   -2.0000
O    20.2594    4.0519   17.0719   15.9994   -2.0000
O    17.5581    3.5657   17.5581   15.9994   -2.0000
O    17.5581    1.3506   19.7732   15.9994   -2.0000
O    20.2594    0.8644   20.2594   15.9994   -2.0000
O    17.0719    4.0519   20.2594   15.9994   -2.0000
O    17.5581    1.3506   17.5581   15.9994   -2.0000
O    14.3706    1.3506   17.5581   15.9994   -2.0000
O    14.8569    4.0519   17.0719   15.9994   -2.0000
O    12.1556    3.5657   17.5581   15.9994   -2.0000
O    12.1556    1.3506   19.7732   15.9994   -2.0000
O    14.8569    0.8644   20.2594   15.9994   -2.0000
O    11.6694    4.0519   20.2594   15.9994   -2.0000
O    12.1556    1.3506   17.5581   15.9994   -2.0000
O    14.3706    6.7531   17.5581   15.9994   -2.0000
O    14.8569    9.4544   17.0719   15.9994   -2.0000
O    12.1556    8.9682   17.5581   15.9994   -2.0000
O    12.1556    6.7531   19.7732   15.9994   -2.0000
O    14.8569    6.2669   20.2594   15.9994   -2.0000
O    11.6694    9.4544   20.2594   15.9994   -2.0000
O    12.1556    6.7531   17.5581   15.9994   -2.0000
```

```
O    19.7732    6.7531   17.5581   15.9994   -2.0000
O    20.2594    9.4544   17.0719   15.9994   -2.0000
O    17.5581    8.9682   17.5581   15.9994   -2.0000
O    17.5581    6.7531   19.7732   15.9994   -2.0000
O    20.2594    6.2669   20.2594   15.9994   -2.0000
O    17.0719    9.4544   20.2594   15.9994   -2.0000
O    17.5581    6.7531   17.5581   15.9994   -2.0000
O    15.3431   12.1556    1.3506   15.9994   -2.0000
O    12.1556   15.3431    1.3506   15.9994   -2.0000
O    14.8569   14.8569    1.8369   15.9994   -2.0000
O    12.1556   12.1556    4.5381   15.9994   -2.0000
O    14.8569   12.6419    4.0519   15.9994   -2.0000
O    12.6419   14.8569    4.0519   15.9994   -2.0000
O    14.8569   14.8569    4.0519   15.9994   -2.0000
O    20.7456   12.1556    1.3506   15.9994   -2.0000
O    17.5581   15.3431    1.3506   15.9994   -2.0000
O    20.2594   14.8569    1.8369   15.9994   -2.0000
O    17.5581   12.1556    4.5381   15.9994   -2.0000
O    20.2594   12.6419    4.0519   15.9994   -2.0000
O    18.0444   14.8569    4.0519   15.9994   -2.0000
O    20.2594   14.8569    4.0519   15.9994   -2.0000
O    20.7456   17.5581    1.3506   15.9994   -2.0000
O    17.5581   20.7456    1.3506   15.9994   -2.0000
O    20.2594   20.2594    1.8369   15.9994   -2.0000
O    17.5581   17.5581    4.5381   15.9994   -2.0000
O    20.2594   18.0444    4.0519   15.9994   -2.0000
O    18.0444   20.2594    4.0519   15.9994   -2.0000
O    20.2594   20.2594    4.0519   15.9994   -2.0000
O    15.3431   17.5581    1.3506   15.9994   -2.0000
O    12.1556   20.7456    1.3506   15.9994   -2.0000
O    14.8569   20.2594    1.8369   15.9994   -2.0000
O    12.1556   17.5581    4.5381   15.9994   -2.0000
O    14.8569   18.0444    4.0519   15.9994   -2.0000
O    12.6419   20.2594    4.0519   15.9994   -2.0000
O    14.8569   20.2594    4.0519   15.9994   -2.0000
O    19.7732   12.1556    6.7531   15.9994   -2.0000
O    20.2594   14.8569    6.2669   15.9994   -2.0000
O    17.5581   14.3706    6.7531   15.9994   -2.0000
O    17.5581   12.1556    8.9682   15.9994   -2.0000
O    20.2594   11.6694    9.4544   15.9994   -2.0000
O    17.0719   14.8569    9.4544   15.9994   -2.0000
O    17.5581   12.1556    6.7531   15.9994   -2.0000
O    14.3706   12.1556    6.7531   15.9994   -2.0000
O    14.8569   14.8569    6.2669   15.9994   -2.0000
O    12.1556   14.3706    6.7531   15.9994   -2.0000
O    12.1556   12.1556    8.9682   15.9994   -2.0000
O    14.8569   11.6694    9.4544   15.9994   -2.0000
O    11.6694   14.8569    9.4544   15.9994   -2.0000
O    12.1556   12.1556    6.7531   15.9994   -2.0000
O    14.3706   17.5581    6.7531   15.9994   -2.0000
O    14.8569   20.2594    6.2669   15.9994   -2.0000
O    12.1556   19.7732    6.7531   15.9994   -2.0000
O    12.1556   17.5581    8.9682   15.9994   -2.0000
O    14.8569   17.0719    9.4544   15.9994   -2.0000
O    11.6694   20.2594    9.4544   15.9994   -2.0000
```

```
O    12.1556   17.5581    6.7531   15.9994   -2.0000
O    19.7732   17.5581    6.7531   15.9994   -2.0000
O    20.2594   20.2594    6.2669   15.9994   -2.0000
O    17.5581   19.7732    6.7531   15.9994   -2.0000
O    17.5581   17.5581    8.9682   15.9994   -2.0000
O    20.2594   17.0719    9.4544   15.9994   -2.0000
O    17.0719   20.2594    9.4544   15.9994   -2.0000
O    17.5581   17.5581    6.7531   15.9994   -2.0000
O    15.3431   12.1556   12.1556   15.9994   -2.0000
O    12.1556   15.3431   12.1556   15.9994   -2.0000
O    14.8569   14.8569   12.6419   15.9994   -2.0000
O    12.1556   12.1556   15.3431   15.9994   -2.0000
O    14.8569   12.6419   14.8569   15.9994   -2.0000
O    12.6419   14.8569   14.8569   15.9994   -2.0000
O    14.8569   14.8569   14.8569   15.9994   -2.0000
O    20.7456   12.1556   12.1556   15.9994   -2.0000
O    17.5581   15.3431   12.1556   15.9994   -2.0000
O    20.2594   14.8569   12.6419   15.9994   -2.0000
O    17.5581   12.1556   15.3431   15.9994   -2.0000
O    20.2594   12.6419   14.8569   15.9994   -2.0000
O    18.0444   14.8569   14.8569   15.9994   -2.0000
O    20.2594   14.8569   14.8569   15.9994   -2.0000
O    20.7456   17.5581   12.1556   15.9994   -2.0000
O    17.5581   20.7456   12.1556   15.9994   -2.0000
O    20.2594   20.2594   12.6419   15.9994   -2.0000
O    17.5581   17.5581   15.3431   15.9994   -2.0000
O    20.2594   18.0444   14.8569   15.9994   -2.0000
O    18.0444   20.2594   14.8569   15.9994   -2.0000
O    20.2594   20.2594   14.8569   15.9994   -2.0000
O    15.3431   17.5581   12.1556   15.9994   -2.0000
O    12.1556   20.7456   12.1556   15.9994   -2.0000
O    14.8569   20.2594   12.6419   15.9994   -2.0000
O    12.1556   17.5581   15.3431   15.9994   -2.0000
O    14.8569   18.0444   14.8569   15.9994   -2.0000
O    12.6419   20.2594   14.8569   15.9994   -2.0000
O    14.8569   20.2594   14.8569   15.9994   -2.0000
O    19.7732   12.1556   17.5581   15.9994   -2.0000
O    20.2594   14.8569   17.0719   15.9994   -2.0000
O    17.5581   14.3706   17.5581   15.9994   -2.0000
O    17.5581   12.1556   19.7732   15.9994   -2.0000
O    20.2594   11.6694   20.2594   15.9994   -2.0000
O    17.0719   14.8569   20.2594   15.9994   -2.0000
O    17.5581   12.1556   17.5581   15.9994   -2.0000
O    14.3706   12.1556   17.5581   15.9994   -2.0000
O    14.8569   14.8569   17.0719   15.9994   -2.0000
O    12.1556   14.3706   17.5581   15.9994   -2.0000
O    12.1556   12.1556   19.7732   15.9994   -2.0000
O    14.8569   11.6694   20.2594   15.9994   -2.0000
O    11.6694   14.8569   20.2594   15.9994   -2.0000
O    12.1556   12.1556   17.5581   15.9994   -2.0000
O    14.3706   17.5581   17.5581   15.9994   -2.0000
O    14.8569   20.2594   17.0719   15.9994   -2.0000
O    12.1556   19.7732   17.5581   15.9994   -2.0000
O    12.1556   17.5581   19.7732   15.9994   -2.0000
O    14.8569   17.0719   20.2594   15.9994   -2.0000
```

```
O    11.6694   20.2594   20.2594   15.9994   -2.0000
O    12.1556   17.5581   17.5581   15.9994   -2.0000
O    19.7732   17.5581   17.5581   15.9994   -2.0000
O    20.2594   20.2594   17.0719   15.9994   -2.0000
O    17.5581   19.7732   17.5581   15.9994   -2.0000
O    17.5581   17.5581   19.7732   15.9994   -2.0000
O    20.2594   17.0719   20.2594   15.9994   -2.0000
O    17.0719   20.2594   20.2594   15.9994   -2.0000
O    17.5581   17.5581   17.5581   15.9994   -2.0000

! This file contains the position, mass and charge of atoms in the
La2Zr2O7 pyrochlore structure.
!   3   (Number of species)
!   704 (Number of atoms:  128 - La, 128 - Zr, 448 - Ox)
```

4. THE PARAMETER INPUT FILE

```
10.0D0      ! cutoff radius (Ang)
7.0D2       ! temperature (K)
10000       ! MD steps
0.5D0       ! step size (fs)
100         ! quench_interval
30          ! quench_times
20          ! write_scalar
```

5. THE HEAT CURRENT ANALYZER PROGRAM

```
program HCACAnalyzer

! Purpose: This program will compute the integral of the Heat Current
! Auto-correlation Function (HCACF), which gives the thermal
! conductivity of the system.  The bounds of this integral will be
! taken from step 1 to each step in the data, and a running average
! will be computed, which should converge to the actual value of the
! thermal conductivity.

IMPLICIT none

! Declare Variables

! curin: The input values of heat current.
! curout: The output values of heat current.
! i: counter
! j: step number of the data
! input: variable name of the input file (HCAC)

    INTEGER i, mavg
    INTEGER j(198000)
    DOUBLE PRECISION heatcorr(198000), k0, jijj, htcdt(198000), &
     thcdt(198000), lambda
    DOUBLE PRECISION volume, temp, kb, tstep
    PARAMETER (volume=1.0092D4, temp=700, kb=8.6173D-5)
    CHARACTER input*5, output*12
```

```
    mavg = 198000

    input='HCAC1'
    output='Conductivity'

    OPEN (UNIT=10,FILE=input,STATUS='OLD')
    READ(10,*) mavg
    READ(10,*) k0
    READ(10,*) tstep
    DO i=1,mavg
      READ (10,*) j(i), heatcorr(i)
    ENDDO

    jijj = 0.0D0
    lambda = 0.0D0

    DO i=1,mavg
      jijj = jijj + heatcorr(i)
      htcdt(i) = 1.6022D6*jijj*tstep/3.0D0/kb/volume/temp/temp
!     (W/m-K)
      lambda = lambda + htcdt(i)
      thcdt(i) = lambda/i
    ENDDO

!   mavg2 = mavg/10
!
!   DO i=1,mavg2
!     interval = 10*i
!     curout(i) = curin(interval)
!   ENDDO

    OPEN (UNIT=11,FILE=output,STATUS='UNKNOWN')
    DO i=1,mavg
      WRITE(11,*) i, htcdt(i), thcdt(i)
    ENDDO
END
```

THIS PAGE INTENTIONALLY LEFT BLANK

LIST OF REFERENCES

[1] G. E. Moore, "Progress in Digital Integrated Electronics," in *Technical Digest 1975, International Electron Devices Meeting, IEEE*, 1975, pp. 11-13.

[2] S. O. Kasap, *Principles of Electronic Materials and Devices,* 3rd Ed. New York: McGraw-Hill, 2006.

[3] S. T. Thornton and A. Rex, *Modern Physics for Scientists and Engineers*. Toronto: Brooks/Cole, 2006.

[4] J. Robertson and R. W. Peacock, "Atomic structure, band offsets, growth and defects at high-K oxide: Si interfaces," *Microelectronic Engineering,* vol. 72, pp. 112-120, 2004.

[5] V. Fiorentini and G. Gulleri, "Theoretical Evaluation of Zirconia and Hafnia as Gate Oxides for Si Microelectronics," *Physical Review Letters,* vol. 89, 266101 pp. 1-4, December 2002.

[6] R. Terki, H. Feraoun, G. Bertrand, and H. Aourang, "Full Potential Linearized Augmented Plane Wave Investigations of Structural and Electronic Properties of Pyrochlore Systems," *Journal of Applied Physics,* vol. 96, pp. 6485-6487, December 2004.

[7] G. Chen, *Nanoscale Energy Transport and Conversion.* New York: Oxford University Press, 2005.

[8] A. Chartier, C. Meis, J. P. Crocombette, L. R. Corrales, and W. J. Weber, "Atomistic Modeling of Displacement Cascades in $La_2Zr_2O_7$ Pyrochlore," *Physical Review B,* vol. 67, 174102 pp. 1-13, May 2003.

[9] M. E. Björketun, C. S. Knee, B. J. Nyman, and G. Wahnström, "Protonic Defects in Pure and Doped $La_2Zr_2O_7$ Pyrochlore Oxide," *Solid State Ionics,* vol. 178, pp. 1642-1647, 2008.

[10] M. E. Wieser, "Atomic Weights of the Elements 2005," *Pure Applied Chemistry*, vol. 78, pp. 2051-2066, 2006.

[11] D. Wolf, P. Keblinski, S. R. Phillpot, and J. Eggebrecht, "Exact Method for the Simulation of Coulombic Systems by Spherically Truncated, Pairwise r^{-1} Summation," *Journal of Chemical Physics*, vol. 110, pp. 8254-8282, May 1999.

[12] D. Wolf, "Reconstruction of NaCl Surfaces from a Dipolar Solution to the Madelung Problem," *Physical Review Letters*, vol. 68, pp. 3315-3318, June 1992.

[13] H. Zhou, D. Yi, Z. Yu, L. Xiao, "Preparation and Thermophysical Properties of CeO_2 doped $La_2Zr_2O_7$ Ceramic for Thermal Barrier Coatings," *Journal of Alloys and Compounds*, vol. 438, pp. 217-221, 2007.

[14] G. Suresh, G. Seenivasan, M.V. Krishnaiah, P. S. Murti, "Investigation of the Thermal Conductivity of Selected Compounds of Gadolinium and Lanthanum," *Journal of Nuclear Materials*, vol. 249, pp. 259 – 261, 1997.

[15] D. C. Rapaport, *The Art of Molecular Dynamics Simulation.* Cambridge: Cambridge University Press, 1997.

[16] Atomic Scale Simulations Class Project, University of Illinois at Urbana-Champaign, "Thermal Conductivity of Argon from the Green-Kubo Method," 2004. [Online]. Available: http://online.physics.uiuc.edu/courses/phys466/Spring07/projects/2004/Team1/index.html.

[17] W. D. Callister, Jr., *Materials Science and Engineering An Introduction*, 7th Ed. York: John Wiley & Sons, 2007.

[18] B. L. Huang, A. J. H. McGaughey, M. Kaviany, "Thermal Conductivity of Metal-Organic Framework 5 (MOF-5): Part I. Molecular Dynamics Simulations," *International Journal of Heat and Mass Transfer*, vol. 50, pp. 393-404, 2007.

[19] W. Huang, E. Humenay, K. Skadron, and M. R. Stan, "The Need for a Full-Chip and Package Thermal Model for Thermally Optimized IC Designs," in Proceedings of the 2005 International Symposium on Low Power Electronics and Designs, 2005, pp. 245-250.

[20] R. B. Shirts, S. R. Burt, and A. M. Johnson, “Periodic Boundary Condition induced Breakdown of the Equipartition Principle and other Kinetic Effects of Finite Sample Size in Classical Hard-Sphere Molecular Dynamics Simulation,” *The Journal of Chemical Physics*, vol. 125, 164102 pp. 1-9, October 2006.

THIS PAGE INTENTIONALLY LEFT BLANK

INITIAL DISTRIBUTION LIST

1. Defense Technical Information Center
   Ft. Belvoir, Virginia

2. Dudley Knox Library
   Naval Postgraduate School
   Monterey, California

3. John Lloyd
   Naval Postgraduate School
   Monterey, California

4. Young Kwon
   Naval Postgraduate School
   Monterey, California